VOLCANICA

Thermobar: An Open-Source Python3 tool for Thermobarometry and Hygrometry

Penny E. Wieser[*†], Maurizio Petrelli[‡], Jordan Lubbers[§], Eric Wieser [¶], Adam Kent[‖], Christy Till[**]

Abstract

We present a new Mineral-Melt Equilibrium tool, Thermobar, written in the open-source language Python3. Thermobar allows pressures, temperatures and melt water contents to be easily calculated using more than 100 popular thermobarometers (e.g., liquid-only, olivine-liquid, olivine-spinel, pyroxene-only, pyroxene-liquid, two pyroxene, feldspar-liquid, two feldspar, amphibole and amphibole-liquid) and hygrometers (e.g., plagioclase-liquid, amphibole). We also provide computationally-fast functions for calculating pressures and temperatures for all possible pairs of phases in equilibrium from a given sample/volcanic center (e.g., cpx-liquid, opx-liquid, two-pyroxene, two-feldspar matching). Finally, the tool contains a number of functions allowing users to calculate equilibrium tests, draw equilibrium diagrams, and use monte-carlo techniques to propagate estimated errors for input parameters into error distributions for H_2O, Pressure and Temperature.

Résumé

French abstract?.

1 Introduction

The pressures and temperatures at which magmas are stored in the Earth's crust and mantle is an important research direction within the field of igneous petrology. For example, the depth of magma storage in arcs plays a vital role in determining the growth, chemical and structural evolution of the Earth's crust (e.g., delamination of mafic cumulates; Lee and Anderson [2015]). Additionally, determining the depths of magma storage beneath active volcanic centres help to inform risk evaluation during periods of volcanic unrest (e.g., distinguishing between unrest signals from the magmatic system indicating that an eruption is imminent, versus signals arising from hydrothermal activity, Pritchard et al. [2019]).

Geophysical methods such as satellite-based interferometric synthetic-aperture radar (InSAR), or a ground-based seismic and Global Positioning System (GPS) networks can place accurate constraints on the locations of magma storage regions beneath active, well-monitored volcanoes. However, many volcanoes have no ground-based monitoring equipment, and even in regions with dense seismic networks, imaging magma reservoirs smaller than <5 km^3, or >10 km depth is extremely challenging (Edmonds et al. [2019]). Additionally, the lack of geodetic and seismic activity at quiescent, dormant or extinct volcanic centres means that these methods cannot be used to deduce magma storage conditions at many of the worlds volcanoes.

An alternative approach uses the concentration of CO_2 and H_2O within small pockets of melt trapped within growing crystals (melt inclusions) to calculate the pressure, and therefore depth within the crust, at which these crystals grew (Wallace et al. [2021]). However, many volcanic systems have very few melt inclusions, or erupt predominantly lava flows where melt inclusions have undergone diffusive loss of H_2O (so record anomalously shallow crystallization pressures; Gaetani et al. [2012])

A third approach, termed (geo)thermobarometry and hygrometry, uses the relationship between the chemistry of erupted crystals and liquids in experiments, and the pressures, temperatures and wa-

[*]Oregon State University
[†]Corresponding author: penny.wieser@gmail.com
[‡]Perugia
[§]Oregon State University
[¶]Cambridge University
[‖]Oregon State University
[**]Arizona State University

ter contents at which these experiments were conducted to deduce the conditions at which natural crystals and melts equilibrated. Barometers rely on the fact that certain chemical equilibrium have significant volume differences between products and reactants, so are sensitive to pressure, while thermometers rely on the fact that other equilibria have significant entropy differences, so are sensitive to temperature (Putirka [2008]). Similarly, hygrometers can be used to estimate the H_2O contents of the melts from which crystal phases grew, because specific mineral-melt phase equilibrium are sensitive to melt H_2O content (e.g., plagioclase-melt hygrometry, Waters and Lange [2015]). Thermobarometry is by far the most broadly applicable method to determine magma storage depths; it can be applied to extrusive rocks which have undergone a range of cooling rates (e.g., tephras, lava flows, unlike melt inclusions), as well as extinct volcanic centres, and intrusive deposits throughout the geological record.

Hundreds of geo-thermobarometers calibrated for common phases present in igneous systems have been calibrated and published over the last few decades, along with detailed reviews of their relative strengths and pitfalls. In particular, the review of Putirka [2008] summarized the most popular thermobarometers, and provided a number of new equations calibrated on experimental data available in LEPR (library of experimental phase relations, Hirschmann et al. [2008]). Alongside this review, K. Putirka released a series of Excel workbooks (currently available at `http://www.fresnostate.edu/csm/ees/faculty-staff/putirka.html`), which are widely used by the community to perform thermobarometry calculations. New thermometers published since this review are available as separate excel spreadsheets (e.g., Pu et al. [2017], Masotta et al. [2013]), excel spreadsheets and Python scripts (e.g., Brugman and Till [2019]), or excel spreadsheets and Matlab scripts (e.g., Waters and Lange [2015]). However, a number of models have no publically-available tool (e.g., Sugawara [2000], Mutch et al. [2016]), although spreadsheets can sometimes be obtained upon request through the authors. This myriad of different calculators, with different input and output structures, means that performing calculations on a variety of different mineral species within a given sample set is very time consuming, and requires users to reformat their data for different input structures in different excel sheets.

Additionally, a number of methods have been developed in recent years which are very difficult to perform in Excel. For example, it is common that only a narrow range of liquid composition will be erupted in any given episode/phase of a volcanic system, while the erupted crystal cargo may be very chemically diverse, having grown from a range of melt compositions undergoing chemical differentiation at depth. The lack of glassy groundmass in many volcanic centres means it is difficult to even characterize the composition of this single "carrier liquid" bringing the crystals to the surface, as bulk analyses techniques such as XRF are sensitive to crystal addition. These pitfalls mean that it is very challenging to identify which minerals grew from which melts in order to perform thermobarometric calculations.

One approach to this problem was developed by Winpenny and Maclennan [2011], who considered all possible pairings of erupted clinopyroxene compositions from a single flow (Borgarhraun) with a compilation of 1000 whole-rock and glass analyses from other icelandic eruptions. They identified pairs in Fe-Mg and trace element partitioning, and used these to calculate pressure and temperature. This method was expanded by Neave and Putirka [2017], who added tests to assess the degree of equilibrium in terms of the EnFs, CaTs and DiHd component as well as Fe-Mg equilibrium. However, these methods require very large numbers of computations to be performed; if users wished to evaluate possible matches for 1000 liquids, and 200 clinopyroxenes, an excel spreadsheet with 200,000 rows would be needed. As well as the tedium of arranging all possible matches in excel, because most clinopyroxene-liquid thermometers are sensitive to pressure and barometers are sensitive to pressure, cells for P and T must be solved iteratively. Inevitably, excel crashes frequently when trying to perform even a subset of these calculations, and many P-T iterations do not converge even if a solution exists. Consequently, Winpenny and Maclennan [2011] performed their calculations in Fortran, and Neave and Putirka [2017] used R. However, these codes are not publically available, and are also relatively slow (the R code takes 30 minutes to perform calculations for a few hundred liquids and clinopyroxenes).

To address the shortage of user-friendly tools for performing popular calculations, we present Thermobar, which is written in the open-source language Python3. Thermobar allows users to quickly perform calculations for more than a hundred popular thermometers, barometers and hygrometers based on inputs provided in an excel spreadsheet requiring minimal formatting of oxide data (because of the utility of the Python Pandas package which can recognise column headings, regardless of their order). As a result of the functionality of the functools module, Thermobar allows users to easily swap between different equations when evaluating pressure or temperature, or iterate different equations to converge towards a solution when neither pressure nor temperature is known. Additionally, we provide a number of advanced functions for assessing equilib-

rium, mineral-liquid and mineral-mineral matching, and monte-carlo error propagation. This tool has been extensively benchmarked against existing calculators, as demonstrated in the supplementary information. A schematic illustration of Thermobar and some of the available functions is shown in Fig. 1.

2 Worked Examples

In this manuscript, we show a number of examples using code snippets to demonstrate that this tool has in-built functions to meet the needs of most users, as well as a large-degree of flexibility for customizing functions. In addition to these examples, we have uploaded a number of jupyter notebooks in the folder "Examples", which contains subfolders with the following names showing example workflows for different phases:

- **Liq_and_Ol_Liq_Thermometry**: Examples calculating T from liquids, and olivine-liquid pairs. Calculations of equilibrium olivine Fo contents from a specific melt composition using a variety of $K_{D,-Mg}$ models. Plotting olivine-melt pairs on Rhodes diagram (where liquid Mg# is plotted against olivine Mg#, with lines for different equilibrium models).
- **Cpx_and_Cpx_Liq_Thermobarometry**: Examples calculating P for known T, T for known P, iteratively solving P and T for clinopyroxene-only and clinopyroxene-liquid pairs. Worked examples are also providd to recreate the clinopyroxene-liquid matching techniques of Scruggs and Putirka [2018] and Gleeson et al. [2020].
- **Opx_and_Opx_Liq_Thermobarometry**: Examples calculating P for known T, T for known P, iteratively solving P and T for orthopyroxene-only and orthopyroxene-liquid pairs. As for Cpx, examples are shown for matching all possible orthopyroxene-liquid pairs filtered by $K_{D,-Mg}$.
- **Two_pyroxene_Thermobarometry**: Examples calculating P for known T, T for known P, iteratively solving P and T for orthopyroxene-clinopyroxene pairs, as well as functions to consider all possible opx-cpx matches filtered by $K_{D,-Mg}$.
- **Amp_and_Amp_Liq _Thermobarometry** Examples calculating P for known T, T for known P, iteratively solving P and T for amphibole-only and amphibole-liquid pairs.
- **Error_propagation**: Examples propagating errors in input parameters for liq-only thermometry, and cpx-liq thermobarometry.

Worked examples can also be found on the Thermobar YouTube Channel https://www.youtube.com/channel/UC7ddceuNnikCdQa_fRHmdXw.

3 Thermobar structure

Thermobar is written as a series of .py files within the folder "Thermobar". After adding the path of this folder to their script, users should type "import Thermobar as pt". After review, Thermobar will also be added to pip, such that users can also install Thermobar using "pip install Thermobar". After import, any function from Thermobar can then be called in the script by typing "pt." followfed by the name of the function. Documentation for each function can be accessed by typing "help(pt.function_name)".

3.1 Python terminology

Thermobar makes extensive use of the functools module, Numpy (Harris et al. [2020]) and Pandas (pandas development team [2020]). Numpy is used to perform operations such as averaging, calculating natural logs and exponentials. Pandas is used to provide flexibility in user inputs, because it allows data to be stored and viewed in a datastructure called dataframes, which are very similar to an excel spreadsheet (e.g., data is stored with column headings) It also means that users can easily import excel spreadsheets where the column order doesn't matter, and Thermobar can identify specific columns (e.g., the SiO_2 content of the melt phase by specifying a column heading as SiO2_Liq). For the plots shown in this paper, the plotting library matplotlib is used. Thus, to follow along with the tutorials, we recommend importing all these packages along with Thermobar at the start of the script (see Fig. 2).

Five main data descriptions are used throughout this paper and the documentation. "strings" are pieces of text, such as selecting the name of which equation to specify in functions using equationP="P_Put2008_eq30". Floats and integers are single numbers, such as selecting P=5 in a function to specify that you want to perform calculations at 5 kbar. Series are a datatype from pandas which can be thought of as a single column of data (e.g. a single column in an excel spreadsheet). Dataframes are also from pandas, and are a collection of panda series (similar to a sheet in Excel). Dictionaries are the highest level data structure described here. In Thermobar, they are frequently used to store multiple panda dataframes, each associated with a specific "key". These dataframes can be thought of as separate sheets in an excel spreadsheet (the dictionary) with the key corresponding to the sheet name. *Eric can you make sure this is technically python-correct without being confusing for beginners).

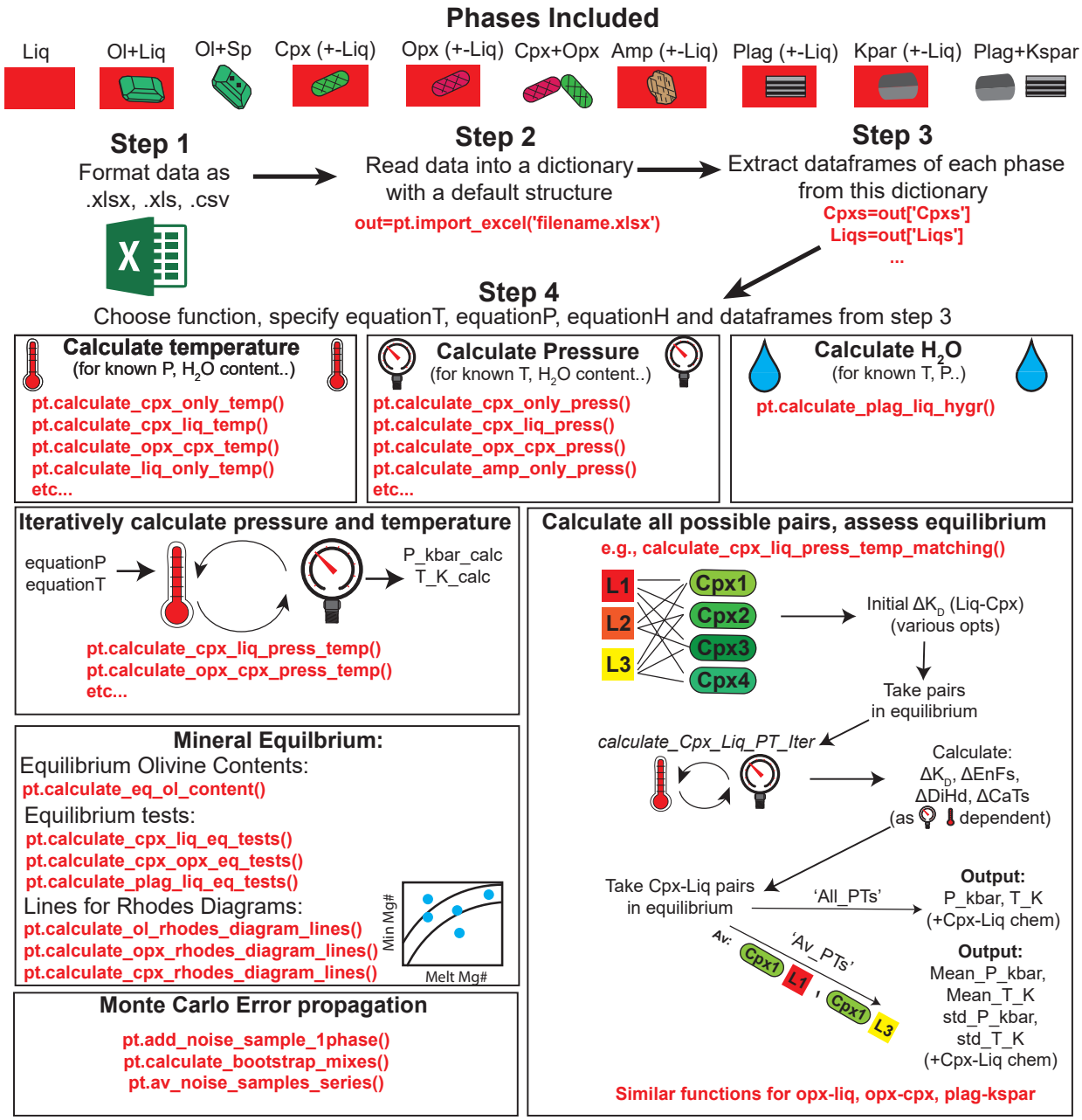


Figure 1: Schematic of Thermobar workflow. Thermobar reads in data supplied from a spreadsheet-type format. The import_excel function formats data into separate dataframes for each phase with a default structure, combined into a single dictionary. Once extracted, these dataframes can be fed into a number of different functions. In addition to simple calculations of T, P and H_2O content, Thermobar allows users to iterate different equations for pressure and temperature, assess all possible matches for pairs of phases, assess mineral equilibrium, and propagate errors.

3.2 Data Input

Users should format their compositional data as an excel spreadsheet (.xlsx, .xls) or a comma separated values (.csv) file, with each analysis having its own row, and oxide components in wt% as columns (Fig. 2). The order of columns doesn't matter, as the python pandas package will identify the column heading regardless of its location. This spreadsheet can be imported into Thermobar using the **import_excel** function, which recognises different phases based on the presence of an underscore followed by a phase identifier in column headings. For example, the column heading SiO2_Liq tells the code that this is the column containing the SiO_2 content of the liquid/melt phase. In order to have the sample ID returned along with the oxides, they should be stored in a column named "Sample_ID_Phase". For example, if the name of an EPMA analysis for an orthopyroxene is opx_pair5_spot1, this should be entered in a column with the heading "Sample_ID_Opx", while the analysis of the touching clinopyroxene EPMA analysis should be in a column with the heading "Sample_ID_Cpx". The full list of identifiers is below:

- Liquid (_Liq)
- Clinopyroxene (_Cpx)
- Orthopyroxene (_Opx)
- Plagioclase (_Plag)
- Alkali feldspar (_Kspar)
- Spinel (_Sp)
- Amphibole (_Amp)

For liquids, Thermobar allows users to specify how they partition Fe between ferrous and ferric iron, because equilibrium tests involving the partitioning of Fe^{2+} and Mg between minerals and melt are sensitive to the proportion of Fe^{3+}. To avoid ambiguity (e.g., in cases where XRF data is reported as Fe_2O_3, but the speciation is unknown vs. times when the user actually knows the proportion of FeO and Fe_2O_3), total FeO contents should be calculated and labelled "FeOt_Liq" (e.g., from EPMA, XRF data). To partition melt Fe between redox states, users ca provide another column called "Fe3Fet_Liq" where the proportion of Fe^{3+} in the liquid is specified. None of the models considered here are sensitive to user-entered Fe redox proportions in phases other than liquid.

Thermobar also has a function **import_excel_err** which recognises columns of the form SiO_2_Cpx_Err as specifying the error for SiO_2 in clinopyroxene. Errors can be entered as absolute values (in wt%) or percentage errors (the error type is specified in the error propagation function).

Both import functions read from the selected excel spreadsheet, and arrange the inputted columns into a dataframe for each mineral phase. To address the fact that many literature datasets have text values (strings) in certain cells (e.g., bdl, n.d, NA, N/A), Thermobar automatically replaces any string in an oxide columns with zeros. If a given column heading is absent, Thermobar also fills this column with zeros. For simplicity, and to create a uniform output structure, if the inputted spreadsheet only contains columns with the headings "_Liq", the returned dataframes for other phases will consistent entirely of zeros.

The dataframes for all recognised phases are joined into a pandas dictionary (called "out" in the example in Fig. 2). The dataframes for each phase are accessed from this output using "dictionaryname['Phasename']" (see Step 2, Fig. 2), where phase names are the same as the column identifiers used in the input spreadsheet, with the addition of an "s". For example, out['Cpxs'] returns the dataframe of Cpxs in Fig. 2. We recommend that these dataframes are inspected before proceeding using the .head() function, which displays the first 5 rows. Column heading for oxides that were not recognised, perhaps due to unusual characters in oxide names, strange decimal points, or spaces in the excel file will be filled with zeros. Inspecting outputs at this stage allows these issues to be identified.

In addition to "recognised" oxide column headings with specified phase identifiers, users can also enter any other columns they wish: for example, for thermometry calculations, users may want to use a pressure derived from other sources, or metadata like latitude, depth within unit etc.. In the example in Fig. 2, pressure is entered in a column labelled "P_MeltInclusions", which might reflect the average pressure calculated from melt inclusions from the same sample. The exact name does not matter; a dataframe is present in the output dictionary called "my_input" which contains all columns from the original spreadsheet.

3.3 Units

Thermobar performs all calculations using temperature in Kelvin, pressure in kbar, and chemistry in wt% for inputs, and the same units for outputs.

3.4 Data Outputs

Thermobar returns two main types of outputs. For simple calculations, e.g., calculating temperature for a given melt composition and pressure, it returns a panda series (a single column of data). For more complicated calculations with more than one output (e.g., pressure and temperature for iterative calculations, or thermobarometers with equilibrium tests), it returns a pandas dataframe. Users can access

any column of values by doing dataframe['column name'] to return a pandas series.

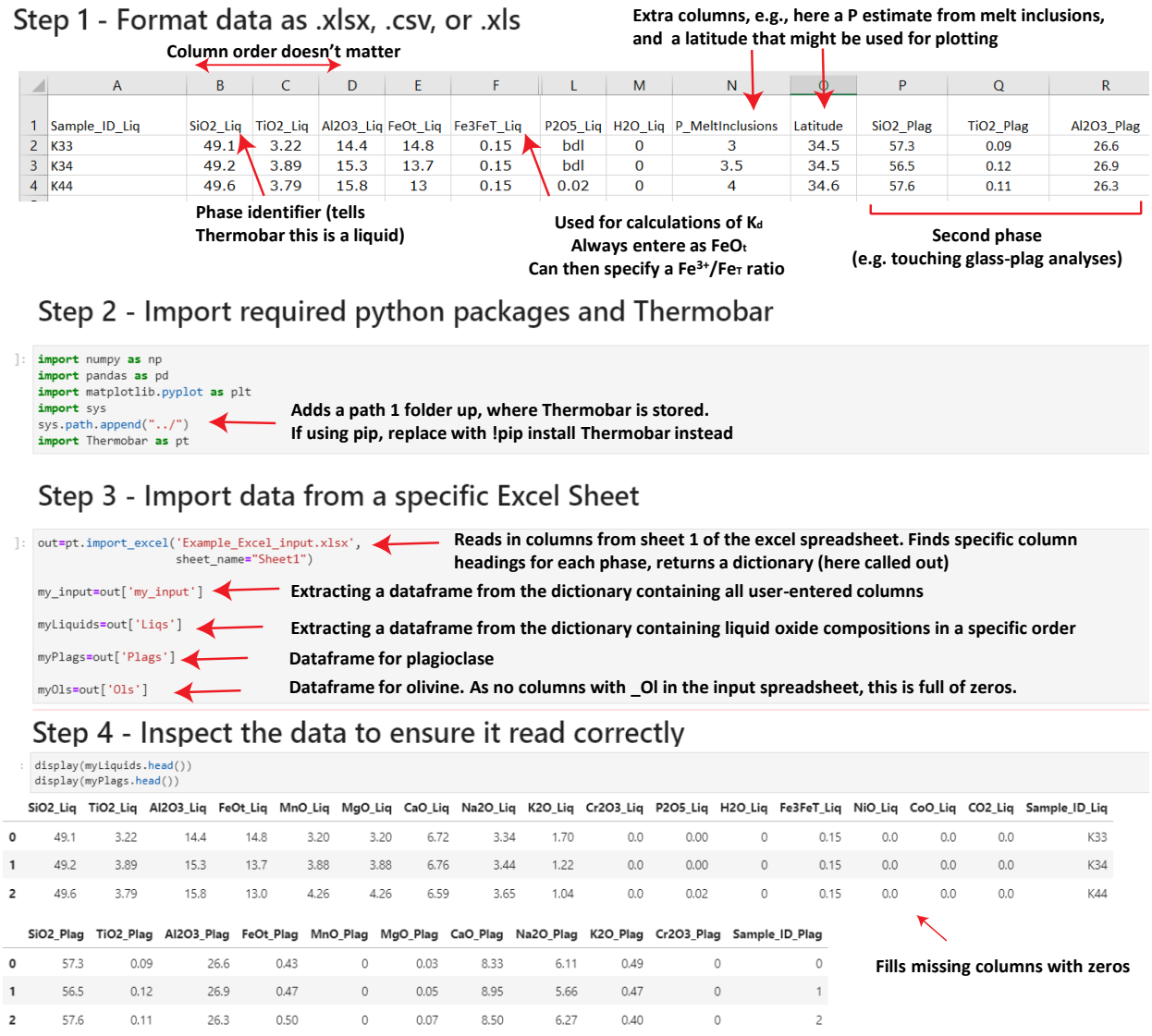
3.5 Warnings

Thermobar contains a number of warnings from the papers of various thermobarometers, which should help to direct users when they are using a model outside its calibration range. These are far from exhaustive, because they rely on the original authors specifying reasonable calibration limits. Some examples:

- If users enter any liquid compositions with SiO_2>68 wt%, and select the cpx-liquid barometer of Neave and Putirka [2017], the code will return the message "Some inputted liquids have SiO2>68 wt%, which exceeds the upper calibration range of the Neave and Putirka (2017) model" (see Fig. 8).
- If users select the clinopyroxene-liquid thermometer of Brugman and Till [2019], the code will check whether: 1) any clinopyroxenes have Mg numbers>0.65, 2) any clinopyroxenes have Al_2O_3>7 wt%, 3) any liquids have SiO_2<70 wt%. If conditions 1 and 3 are met, a warning message will be returned: "Some inputted Cpx compositions have Mg#>0.65, some inputted Liq compositions have SiO2<70 wt%, which is outside the recommended calibration range of Brugman and Till (2019)"

4 Available functions and equations

The thermometers, barometers, and hygrometers available in Thermobar, along with the relevant functions and names in Thermobar, are summarized in Tables 3-6. Thermometry equations are selected using "equationT=", barometry equations using "equationP=" and hygrometry equations using "equationH=". These tables also indicate whether thermometers are dependent on pressure, or whether barometer are sensitive to temperature. If a pressure or temperature is required but has not been specified, Thermobar will return an error advising users a required input is missing. These tables also indicate which equations are sensitive to the H_2O content of the liquid. By default, if a column with H_2O_Liq is not provided in the input spreadsheet, it is assumed H_2O=0, and no error is returned when users select a water-dependent equation. As demonstrated in the code snippets below, the concentration of H_2O in the liquid can easily be overwritten within the functions themselves.



	A	B	C	D	E	F	L	M	N	O	P	Q	R
1	Sample_ID_Liq	SiO2_Liq	TiO2_Liq	Al2O3_Liq	FeOt_Liq	Fe3FeT_Liq	P2O5_Liq	H2O_Liq	P_MeltInclusions	Latitude	SiO2_Plag	TiO2_Plag	Al2O3_Plag
2	K33	49.1	3.22	14.4	14.8	0.15	bdl	0	3	34.5	57.3	0.09	26.6
3	K34	49.2	3.89	15.3	13.7	0.15	bdl	0	3.5	34.5	56.5	0.12	26.9
4	K44	49.6	3.79	15.8	13	0.15	0.02	0	4	34.6	57.6	0.11	26.3

	SiO2_Liq	TiO2_Liq	Al2O3_Liq	FeOt_Liq	MnO_Liq	MgO_Liq	CaO_Liq	Na2O_Liq	K2O_Liq	Cr2O3_Liq	P2O5_Liq	H2O_Liq	Fe3FeT_Liq	NiO_Liq	CoO_Liq	CO2_Liq	Sample_ID_Liq
0	49.1	3.22	14.4	14.8	3.20	3.20	6.72	3.34	1.70	0.0	0.00	0	0.15	0.0	0.0	0.0	K33
1	49.2	3.89	15.3	13.7	3.88	3.88	6.76	3.44	1.22	0.0	0.00	0	0.15	0.0	0.0	0.0	K34
2	49.6	3.79	15.8	13.0	4.26	4.26	6.59	3.65	1.04	0.0	0.02	0	0.15	0.0	0.0	0.0	K44

	SiO2_Plag	TiO2_Plag	Al2O3_Plag	FeOt_Plag	MnO_Plag	MgO_Plag	CaO_Plag	Na2O_Plag	K2O_Plag	Cr2O3_Plag	Sample_ID_Plag
0	57.3	0.09	26.6	0.43	0	0.03	8.33	6.11	0.49	0	0
1	56.5	0.12	26.9	0.47	0	0.05	8.95	5.66	0.47	0	1
2	57.6	0.11	26.3	0.50	0	0.07	8.50	6.27	0.40	0	2

Figure 2: Guide to data input. Step 1: Format data into a spreadsheet with oxide names followed by _phase. The order of columns doesn't matter, and other columns can also be included in the input (e.g., estimates of pressure and temperature, additional sample information etc). Step 2: Thermobar is imported, along with numpy, pandas and matplotlib. Step 3: The **import_excel** function extracts data from this spreadsheet into a set of dataframes with specific a specific column order. The function returns a dictionary (here called "out") where all these dataframes are stored with keys corresponding to different phases. For example, the dataframe of liquids is extracted from this dictionary using the key "Liqs". All dictionary keys correspond to the phase identifiers used for inputs with an added "s". If the input doesn't have specific column headings (E.g., no _Ol, _Kspar), the dataframe for this phase will be filled with zeros. Step 4. Dataframes for each phase are inspected to check that the spreadsheet has been read in correctly. In particular, strange characters for subscripts in the column heading may cause a column to be full of zeros in the output.

Reference	Name in ThermoBar	Pressure-dependent?	H_2O-dependent?
Liquid-only thermometry. Function: "calculate_liq_only_temp"			
Olivine-Sat Liquids			
Putirka (2008)	T_Put2008_eq13	No	No
	T_Put2008_eq14	No	**Yes**
	T_Put2008_eq15	**Yes**	**Yes**
Helz & Thornber, (1987)	T_Helz1987_MgO	No	No
Montierth (1995)	T_Montierth1995_MgO	No	No
Sugawara (2000)	T_Sug2000_eq1	No	No
	T_Sug2000_eq3_ol	**Yes**	No
	T_Sug2000_eq6a	**Yes**	No
	T_Sug2000_eq6a_H7a	**Yes**	**Yes**
Beattie (1993)	T_Beatt93_ BeattDMg	**Yes**	No
	T_Beatt93_BeattDMg_HerzCorr	**Yes**	No
Putirka (2008)	T_Put2008_eq19_BeattDMg	**Yes**	No
	T_Put2008_eq21_BeattDMg	**Yes**	**Yes**
	T_Put2008_eq22_BeattDMg	**Yes**	**Yes**
Cpx-Sat Liquids			
Putirka (2008)	T_Put2008_eq34_cpx_sat	**Yes**	**Yes**
Putirka (1999)	T_Put1999_cpx_sat	**Yes**	No
Sugawara (2000)	T_Sug2000_eq3_cpx	**Yes**	No
	T_Sug2000_eq3_pig	**Yes**	No
	T_Sug2000_eq6b	**Yes**	No
	T_Sug2000_eq6b_H7b	**Yes**	**Yes**
Opx-Sat Liquids			
Putirka (2008)	T_Put2008_eq28b_opx_sat	**Yes**	**Yes**
Sugawara (2000)	T_Sug2000_eq3_opx	**Yes**	No
Beattie (1993)	T_Beatt1993_opx	**Yes**	No
Amp-Sat Liquids			
Putirka (2008)	T_Put2016_eq3_amp_sat	No	Yes*
Molina (2015)	T_Molina2015_amp_sat	No	No
Fspar-Sat Liquids			
Putirka (2005)	T_Put2005_eqD_plag_sat	**Yes**	**Yes**
Putirka (2008)	T_Put2008_eq26_plag_sat	**Yes**	**Yes**
	T_Put2008_eq24c_kspar_sat	**Yes**	**Yes**
Ol-Cpx-Plag Sat Liquids			
Putirka (2008)	T_Put2008_eq16	**Yes**	No

Figure 3: Summary of equations for liquid-only thermometry. *Note, Putirka (2016) equation 3 doesn't contain a H_2O term, but is H_2O-sensitive because liquid cation fractions are calculated on a hydrous basis. Equations from: Putirka [2008], Sugawara [2000], Montierth et al. [1995], Helz and Thornber [1987], Beattie [1993], Herzberg and O'hara [2002], Putirka [1999], Molina et al. [2015], Putirka [2016]

Reference	Name in ThermoBar	Temperature-dependent?	Pressure-dependent?	H_2O-Dependent?
Clinopyroxene-Liquid Barometry. *Function "calculate_cpx_liq_press"*				
Putirka (1996)	P_Put1996_eqP1	**Yes**		No
	P_Put1996_eqP2	**Yes**		No
Putirka (2003)	P_Put2003	**Yes**		No
Putirka (2008)	P_Put2008_eq30	**Yes**		**Yes**
	P_Put2008_eq31	**Yes**		**Yes**
	P_Put2008_eq32c	**Yes**		**Yes**
Masotta et al. (2013) *recalibration of Putirka eqs. for alkali systems*	P_Mas2013_eqPalk1	**Yes**		No
	P_Mas2013_eqPalk2	**Yes**		No
	P_Mas2013_eqalk32c	**Yes**		**Yes**
Masotta et al. (2013)	P_Mas2013_Palk2012	No		**Yes**
Neave & Putirka (2017)	P_Neave2017	**Yes**		No
Clinopyroxene-Liquid Thermometry. *Function "calculate_cpx_liq_temp"*				
Putirka (1996)	T_Put1996_eqT1		No	No
	T_Put1996_eqT2		**Yes**	No
Putirka (1999)	T_Put1999		**Yes**	No
Putirka (2003)	T_Put2003		**Yes**	No
Putirka (2008)	T_Put2008_eq33		**Yes**	**Yes**
Masotta et al. (2013) *Recalibration of Putirka eqs. for alkali systems*	T_Mas2013_eqTalk1		No	No
	T_Mas2013_eqTalk2		**Yes**	No
	T_Mas2013_eqalk33		**Yes**	**Yes**
Masotta et al. (2013)	T_Mas2013_Talk2012		No	**Yes**
Brugman & Till (2019)	T_Brug2019		No	No
Clinopyroxene-only Barometry. *Function "calculate_cpx_only_press"*				
Putirka (2008)	P_Put2008_eq32a	**Yes**		No
	P_Put2008_eq32b	**Yes**		**Yes**
Clinopyroxene-only Thermometry. *Function "calculate_cpx_only_temp"*				
Putirka (2008)	T_Put2008_eq32d		**Yes**	No
	T_Put2008_eq32d_subsol		**Yes**	No
Orthopyroxene-Liquid Barometry. *Function "calculate_opx_liq_press"*				
Putirka (2008)	P_Put2008_eq29a	**Yes**		**Yes**
	P_Put2008_eq29b	**Yes**		**Yes**
Putirka Supplement New "Global" calibrations	P_Put_Global_Opx	No		No
	P_Put_Felsic_Opx	No		No
Orthopyroxene-Liquid Thermometry. *Function "calculate_opx_liq_temp"*				
Putirka (2008)	T_Put2008_eq28a		**Yes**	**Yes**
	T_Put2008_eq28b_opx_sat		**Yes**	**Yes**
Orthopyroxene-only Barometry. Function "calculate_opx_only_press"				
Putirka (2008)	P_Put2008_eq29c	**Yes**		No
Orthopyroxene-Clinopyroxene Barometry. *Function "calculate_cpx_opx_press"*				
Putirka (2008)	P_Put2008_eq38	No		No
	P_Put2008_eq39	**Yes**		No
Orthopyroxene-Clinopyroxene Thermometry. *Function "calculate_cpx_opx_press"*				
Putirka (2008)	T_Put2008_eq36		**Yes**	No
	T_Put2008_eq37		**Yes**	No
Brey and Kohler (1990)	T_Brey1990		**Yes**	No
Wood and Banno (1973)	T_Wood1973		No	No
Wells (1977)	T_Wells1977		No	No
Other Functions				
calculate_cpx_liq_press_temp(), calculate_cpx_only_press_temp(), calculate_cpx_opx_press_temp() Iteratively solves P and T for cpx-liq pairs/cpx-only/cpx-opx using an equation for P and an equation for T				
calculate_cpx_liq_press_temp_matching(), calculate_cpx_opx_press_temp_matching(): Calculates P and T for all possible cpx-liquid/cpx-opx pairs (with user-selected options for equilibrium criteria)				

Figure 4: Summary of equations for pyroxene thermobarometry. From: Putirka [2008], BREY and Köhler [1990], Wells [1977], Wood and Banno [1973], Putirka et al. [1996], Putirka et al. [2003], Beattie [1993], Neave and Putirka [2017], Brugman and Till [2019]. The "Global" and "Felsic" barometers are from the spreadsheets available at http://www.fresnostate.edu/csm/ees/faculty-staff/putirka.html. These equations are particularly-suited to low pressure, low-Al orthopyroxenes where existing thermometers return a numerical error.

Reference	Name in Thermobar	Temperature -dependent?	Pressure-dependent?	H_2O-dependent?
Amphibole-Liquid Barometry. *Function "calculate_amp_liq_press"*				
Putirka (2016)	P_Put2016_eq7a	No		**Yes**
	P_Put2016_eq7b	No		Yes*
	P_Put2016_eq7c	No		Yes*
Amphibole-Liquid Thermometry. *Function "calculate_amp_liq_temp"*				
Putirka (2016)	T_Put2016_eq4b		No	**Yes**
	T_Put2016_eq4a_amp_sat		No	Yes*
	T_Put2016_eq9		No	Yes*
Amphibole-only Barometry. *Function "calculate_amp_only_press"*				
Ridolfi and Renzulli (2012)	P_Ridolfi2012_1a	No		No
	P_Ridolfi2012_1b	No		No
	P_Ridolfi2012_1c	No		No
	P_Ridolfi2012_1d	No		No
	P_Ridolfi2012_1e	No		No
Ridolfi et al. (2010)	P_Ridolfi2010	No		No
Hammerstrom & Zen (1986)	P_Hammerstrom1986_eq1	No		No
	P_Hammerstrom1986_eq2	No		No
	P_Hammerstrom1986_eq3	No		No
Hollister et al. (1987)	P_Hollister1987	No		No
Johnson & Rutherford (1989)	P_Johnson1989	No		No
Blundy et al. (1990)	P_Blundy1990	No		No
Schmidt (1992)	P_Schmidt1992	No		No
Anderson & Smith, 1995	P_Anderson1995	**Yes**		No
Amphibole-only Thermometry. *Function "calculate_amp_only_temp"*				
Putirka (2016)	T_Put2016_eq5		No	No
	T_Put2016_eq6		No	No
	T_Put2016_SiHbl		No	No
	T_Put2016_eq8		**Yes**	No
Ridolfi and Renzuli, 2012	T_Ridolfi2012		**Yes**	No
Other Functions				
***calculate_amp_liq_press_temp*:** Iteratively solves P and T for liquid-amphibole pairs using an equation for pressure, and an equation for temperature. **calculate_amp_only_press_temp:** Iteratively solves P and T for amphibole compositions using an equation for pressure, and an equation for temperature.				

Figure 5: Summary of equations for amphibole thermobarometry. From: Putirka [2016], Mutch et al. [2016], Ridolfi and Renzulli [2012], Ridolfi et al. [2010], Hammarstrom and Zen [1986], Hollister et al. [1987], Johnson [1988], Blundy and Holland [1990], Schmidt [1992], Anderson and Smith [1995].

Olivine- Thermometers

Reference	Name in ThermoBar		Pressure-dependent?	H_2O-dependent?
Olivine-Liquid thermometry. *Function "calculate_ol_liq_temp"*				
Putirka (2008)	T_Put2008_eq19		**Yes**	No
	T_Put2008_eq21		**Yes**	**Yes**
	T_Put2008_eq22		**Yes**	**Yes**
Beattie (1993)	T_Beatt93_ol		**Yes**	No
	T_Beatt93_ol_HerzCorr		**Yes**	No
Sisson and Grove (1992)	T_Sisson1992		**Yes**	No
Pu et al. (2017)	T_Pu2017		No	No
Pu et al. (2021)	T_Pu2021		**Yes**	No
Olivine-Spinel thermometry. *Function "calculate_ol_sp_temp"*				
Coogan et al. (2014)	T_Coogan2014		No	No
Wan et al. (2008)	T_Wan2008		No	No

Feldspar Thermometers, Barometers and Hygrometers

Reference	Name in ThermoBar	Temperature-dependent?	Pressure-dependent?	H_2O-dependent?
Plagioclase-Liquid thermometry. *Function "calculate_fspar_liq_temp"*				
Putirka (2008)	T_Put2008_eq23		**Yes**	**Yes**
	T_Put2008_eq24a		**Yes**	**Yes**
Plagioclase-Liquid Barometry. *Function "calculate_fspar_liq_press"*				
Putirka (2008)	P_Put2008_eq25	**Yes**		No
Alkali Feldspar-Liquid thermometry. *Function "calculate_fspar_liq_temp"*				
Putirka (2008)	T_Put2008_eq24b		**Yes**	No
Plagioclase-Alkali Feldspar thermometry. Function "calculate_plag_kspar_temp"				
Putirka (2008)	T_Put2008_eq27a		**Yes**	No
	T_Put2008_eq27b		**Yes**	No
	T_Put_Global_2Fspar		**Yes**	No
Plagioclase-Alkali Feldspar thermometry. Function "calculate_plag_kspar_temp_matching"				
Putirka (2008)	H_Put2008_eq25b	**Yes**	**Yes**	
Putirka (2005)	H_Put2005_eqH	**Yes**	No	
Waters and Lange (2015)	H_Waters2015	**Yes**	**Yes**	
Other Functions				
Iterative solving of pressure and temperature: **calculate_fspar_liq_press_temp:** Iteratively solves P and T for fspar-liq pairs using an equation for pressure, and an equation for temperature **Matching all possible pairs** **calculate_plag_kspar_temp_matching:** Calculates P and T for all possible plag-kspar pairs (with user-selected options for equilibrium criteria)				

Figure 6: Summary of equations for feldspar thermobarometry and olivine-liquid and olivine-spinel thermometry. For olivine-liquid and olivine-spinel, the equations are from: Putirka [2008], Beattie [1993], Herzberg and O'hara [2002], Sisson and Grove [1993], Pu et al. [2021], Pu et al. [2017], Wan et al. [2008], Coogan et al. [2014]. For feldspars, the equations are from: Putirka [2008], Putirka [2005], Waters and Lange [2015].

5 Single-phase thermometers and Barometers

Thermobar contains a number of thermometers and barometers which are based on the composition of a single phase:

- Liquid-only thermometry
- Clinopyroxene-only thermometry and barometry
- Orthopyroxene-only barometry
- Amphibole-only thermometry and barometry

We discuss some examples for liquid-only thermometry, but the flexibility of function inputs is the same for other single-phase thermobarometers (simply swapping all instances of the word liq for the lower-case name of the other phase).

5.1 Liquid-only thermometers

Liquid-only thermometers vary widely in complexity. For example, the thermometer of Helz and Thornber [1987] calculates the temperature of a liquid based solely on the liquid MgO content, while equation 15 of Putirka [2008] uses the MgO, FeO, Na_2O, K_2O, H_2O content and Mg# of the liquid, as well as an estimate of the pressure. For liquid-only thermometers, most equations calculate the temperature of the liquid, but equations in Thermobar with names ending with "_sat" calculate the temperature at which a liquid is saturated in a specific phase. For example, equation 34 of Putirka [2008] calculates the temperature at which clinopyroxene would saturate in the liquid (termed the saturation surface).

Several liquid-only themometers are adapted from olivine-liquid thermometers, where the D_{Mg} term that would be calculated from olivine-liquid pairs is replaced with a theoretical value of D_{Mg}, calculated from the liquid composition using the model of Beattie (1993). These equations are indicated with _BeattDMg in their name, and are particularly useful because many olivine crystals are not in Fe-Mg equilibrium with their co-erupted carrier melts (see section *6.0.2*), so it is difficult to select an olivine and liquid composition in equilibrium.

Calculations using liquid-only thermometers are performed using the function **calculate_liq_only_temp**. The required inputs are a dataframe of liquid compositions, as well as an "equationT". For example, for a pandas dataframe of liquids named "myLiquids" as in Fig. 2, temperature using the MgO thermometer of Helz and Thornber [1987] would be calculated as follows:

```
Temp_HT87=pt.calculate_liq_only_temp(liq_comps=myLiquids,
        equationT="T_Helz1987_MgO")
```

For equation 15 of Putirka [2008], Thermobar returns an error because this equation is P-sensitive:

```
Temp_P2008_eq15=pt.calculate_liq_only_temp(Liq_Comps=myLiquids,
        equationT="T_Put2008_eq15")
```

```
Exception: You've selected a P-dependent function,
please pass an option for P (see help for more detail)
```

There are a number of ways to specify pressure. Firstly, a constant value of pressure can be specified for all liquids (here, P=5 kbar):

```
Temp_P2008_eq15=pt.calculate_liq_only_temp(liq_comps=myLiquids,
        equationT="T_Put2008_eq15", P=5)
```

Alternatively, if the input spreadsheet contains a column for P in kbar with different values for different liquids, P can be set to the values found in the column labelled"P_input" in the excel spreadsheet:

```
Temp_eq15_in=pt.calculate_liq_only_temp(liq_comps=myLiquids,
        equationT="T_Put2008_eq15", P=my_input['P_input'])
```

Any name of any column in the inputted spreadsheet can be specified in this way, as all columns are stored in the dataframe "my_input" returned from the **import_excel** function (See Fig. 2).

Alternatively, if the pressure isn't known, setting P="Solve" will return a partial function. This can then be evaluated at any particular P by typing the name of this partial function and the pressure in kbar in brackets.

```
Temp_P2008_eq15_par=pt.calculate_liq_only_temp(liq_comps=myLiquids,
        equationT="T_Put2008_eq15", P="Solve")
Temp_P2008_eq15_par(3)
```

For example, here the partial function is defined, and then evaluated at 3 kbar. For large numbers of calculations, this is more efficient than running the function again at a different pressure (because cation fractions etc. don't need to be recalculated when the pressure is changed).

Once calculations have been performed in Thermobar, there are a number of ways to save calculations to an excel workbook to interact with them outside of Python. For example, to save the temperatures alongside the liquid compositions, it is easiest to first make a copy of the original dataframe using the .copy() function. This means that the original is still preserved in the script for further calculations. Then, panda series generated by each calculation can be added onto this dataframe using the pandas .insert() function. Users need to specify a number for which position they want this new column in, as well as the name of the column.

```
Liquid_T_out=myLiquids.copy()
Liquid_T_out.insert(0, "T HT87", Temp_HT87)
Liquid_T_out.insert(1, "T Peq15", Temp_P2008_eq15)
Liquid_T_out.to_excel('FileName.xlsx')
```

In this example, we have saved the calculations from Helz and Thornber [1987] to the 1st column of the dataframe (python numbering starts from zero), and calculations from Putirka [2008] equation 15 to the second column.

Some liquid-only thermometers are also sensitive to melt H_2O content (see Fig. 3), which is of-

ten poorly constrained in volcanic systems where there is no rapidly quenched tephra suitable for melt inclusion analyses. By default, Thermobar will read H_2O contents from the H2O_Liq column of the input spreadsheet. If the input spreadsheet has no column for H_2O, this column is filled with zeros. Input water contents can be overwritten when calling the function by specifying H2O_Liq=..., allowing an easy way to investigate the effect of uncertain H_2O contents on temperatures. H2O_Liq can be set within the function as a constant value (e.g., 6 wt% in the example here)).

```
Temp_P2008_eq15_6H=pt.calculate_liq_only_temp(liq_comps=myLiquids,
        equationT="T_Put2008_eq15", P=5, H2O_Liq=6)
```

```
Water content replaced with that from H2O_Liq
```

As for pressure, H_2O can also be set to the value of any column in the input spreadsheet using H_2O_Liq=my_input['column name'].

5.2 Mineral-only thermometers and barometers

Mineral-only thermometers and barometers are implemented in a very similar way to liquid thermometers.

For example, to calculate amphibole-only pressures using the barometer of Mutch et al. [2016]:

```
pt.calculate_amp_only_press(amp_comps=myAmps,
   equationP="P_Mutch2016")
```

Where myAmps is dataframe of amphibole compositions from the **import_excel** function.

To calculate clinopyroxene-only pressure using equation 32b of Putirka (2008):

```
pt.calculate_cpx_only_press(cpx_comps=myCpxs,
   equationP="P_Put2008_eq32b", T=1400)
```

Where myCpxs is dataframe of clinopyroxene compositions from the **import_excel** function, and 1400 is the temperature in Kelvin at which to perform calculations.

If neither pressure nor temperature is known, an equation can be selected for both pressure and temperature:

```
pt.calculate_cpx_only_press_temp(cpx_comps=myCpxs,
equationP="P_Put2008_eq32b", equationT="T_Put2008_eq32d")
```

5.3 Iterative calculations

Unlike for experimental studies, in natural systems, it is unlikely that either temperature or pressure is known. Thus, Thermobar contains functions for to iterate towards a solution using an equation for pressure and an equation for temperature. The names of these function are adapted from those discussed above by adding the ending _press_temp (e.g., calculate_cpx_only_press_temp). By default, these functions start with T=1300 K, and input this temperature in the selected barometer to calculate a pressure. This calculated pressure is then entered into the selected thermometer, and this process is repeated for 30 iterations. If necessary, users can overwrite both the initial T and number of iterations, although in a multitude of tests, this method converged on a solution identical to the excel iteration used in the spreadsheets of K. Putirka.

6 Two-phase thermometers and Barometers

The following thermometers, barometers and hygrometers are based on equilibrium between two phases. The application of these functions generally require more thought from the user. In an ideal scenario, calculations should be performed on phases which have a clear textural relationship, such as measurements of spinels trapped within a specific olivine crystal (Matthews et al. [2016]), or measurements of touching clinopyroxene-orthopyroxene pairs (Walker et al. [2013]). However, in many natural samples, this is simply not possible. For example, disaggregation of crystals during transport and eruption mean that it is very commmon that erupted lavas and tephra samples have few, or no touching pairs of crystals. Even if crystals are touching, there is no guarantee that they are in chemical equilibrium, as crystals with different histories can be aggregated into clusters by flow within volcanic conduits (Wieser et al. [2019b], Culha et al. [2020]).

Thermobarometers which rely on the equilibrium between a liquid and crystal phase (rather than 2 crystal phases) are particularly problematic. Generally, only a narrow range of liquid composition will be erupted in any given phase of an eruption, while the erupted crystal cargo may be chemically diverse, having grown from a range of melt compositions undergoing chemical differentiation at depth. In many volcanic centres, the lack of glassy groundmass means it is difficult to even characterize the composition of this single "carrier liquid" bringing the crystals to the surface, as bulk analyses techniques such as XRF are sensitive to crystal addition. These pitfalls mean that it is very difficult to identify meaningful mineral-melt pairs in many volcanic systems.

In Thermobar, we provide a number of functions implementing tools proposed in the literature to help users with these less-than-optimal (but common) scenarios. Firstly, we present algorithms which allow users to consider all possible matches between measured phases (e.g., assessing all possible liquid and pyroxene pairs, or all possible pairs of orthopyroxenes and clinopyroxenes). Alongside these matching algorithms, a number of equilibrium tests are performed, to allow users to assess which pairs of phases are likely out of equilibrium. Where relevant, the tests implemented for each thermobarometer are discussed below.

6.0.1 *Olivine-Spinel Thermometry*

Thermobar includes the olivine-spinel thermometers of Wan et al. [2008] and Coogan et al. [2014] (Fig. 6), which are both pressure-independent. Users should format a spreadsheet where each row contains an olivine compositions (column headings SiO2_Ol...) and a spinel composition (SiO2_Sp...). After using the **import_excel** function, these thermometers are called using the function **calculate_ol_sp_temp**:

```
pt.calculate_ol_sp_temp(liq_comps=myLiquids,
                        sp_comps=mySps, equationT="T_Wan2008")
```

Where myOls is a dataframe of olivine compositions, mySps is a dataframe of spinel compositions, and the thermometer is from Wan et al. [2008]. To our knowledge, there are no available equilibrium tests, although the fact that spinels are often incorperated within olivines, along with the slow diffusion rate of Al, means disequilibrium is unlikely to be an issue.

6.0.2 *Olivine-Liquid Thermometry*

Unlike olivine-spinel thermometry, olivine-liquid thermometry is highly susceptible to issues involving disequilibrium. This is because olivine crystals are commonly "antecrystic", being brought to the surface in chemically-unrelated melts (Wieser et al. [2019a]; Balta et al. [2013]). Thus, it is vital to calculate the degree of equilibrium for olivine-liquid pairs to assess the accuracy of thermometric estimates. The most common way to assess olivine-melt equilibrium examines the partitioning of Fe-Mg between these two phases ($K_{D,\,Fe-Mg}^{Ol-Liq}$). Thermobar contains a number of functions to aid with these comparisons.

Firstly, the function **calculate_eq_ol_content** calculates the equilibrium olivine forsterite content for a set of liquid compositions. Three models for predicting $K_{D,\,Fe-Mg}^{Ol-Liq}$ equilibrium are included. Specifying Kd_model="Roeder1970" uses $K_{D,\,Fe-Mg}$ =0.3±0.03 following Roeder and Emslie [1970], Kd_model="Matzen2011" uses $K_{D,\,Fe-Mg}$ =0.34±0.012 following Matzen et al. [2011].By default, this function uses the value of "Fe3FeT_Liq" in the user input.

For example, the following code calculates the equilibrium olivine content using the model of Roeder and Emslie [1970]:

```
Eq_ol_Roeder=pt.calculate_eq_ol_content(liq_comps=myLiquids,
                Kd_model="Roeder1970")
```

The panda dataframe returned by the function has column headings corresponding to the equilibrium forsterite content for $K_{D,\,Fe-Mg}$=0.3 (preferred value), 0.33 (+ 1σ), and 0.27 (- 1σ):

	Eq Fo (Roeder, Kd=0.3)	Eq Fo (Roeder, Kd=0.33)	Eq Fo (Roeder, Kd=0.27)
0	0.616254	0.593479	0.640846
1	0.677781	0.656623	0.700347
2	0.708781	0.688724	0.730041

This default can be overwritten by specifying a value for "Fe3FeT_Liq" in the function:

```
pt.calculate_eq_ol_content(liq_comps=myLiquids,
                Kd_model="Roeder1970", Fe3FeT_Liq=0.2)
```

Unlike the fixed $K_{D,\,Fe-Mg}$ values of Roeder and Emslie [1970] and Matzen et al. [2011], the model of Toplis [2005] calculates $K_{D,\,Fe-Mg}$ as a function of liquid composition, pressure, temperature, and olivine forsterite content. Thermobar provides three ways to use this model. First, the olivine forsterite content can be specified along with pressure, temperature, and liquid compositions:

```
pt.calculate_eq_ol_content(liq_comps=myLiquids,
                Kd_model="Toplis2005", P=2, T=1373.1, ol_fo=0.82)
```

	Kd (Toplis, input Fo)	Eq Fo (Toplis, input Fo)
0	0.282841	0.630083

This returns a panda dataframe where the first column is the equilibrium $K_{D,\,Fe-Mg}$ calculated using Toplis [2005], and the second column is the equilibrium olivine forsterite content. Second, the olivine composition could be specified using ol_comps=" ". However, specifying olivine forsterite content to calculate an equilibrium forsterite content is somewhat circular. If olivine compositions or a forsterite content are not entered into the function, Thermobar will iterate by first calculating a $K_{D,\,Fe-Mg}$ for Fo=0.95, then use this $K_{D,\,Fe-Mg}$ to calculate an equilibrium Fo content, and then inputting that Fo content into a new calculation for $K_{D,\,Fe-Mg}$ (over 20 iterations).

```
pt.calculate_eq_ol_content(liq_comps=myLiquids,
                Kd_model="Toplis2005", P=2, T=1373.1)
```

	Kd (Toplis, Iter)	Eq Fo (Toplis, Iter)
0	0.316984	0.603150
1	0.306317	0.673213

Users can also enter Kd_model="All" to get calculates for all 3 models (Toplis, Matzen and Roeder).

As with olivine-spinel thermometry, the default way to calculate olivine-liquid temperatures in Thermobar is to prepare an excel spreadsheet with each row containing an olivine and liquid composition. For all olivine-liquid thermometers except that of Pu et al. [2017], a pressure needs to be specified (as in section *6.0.2*). For example, temperatures can be calculated using equation 21 of Putirka [2008] at 5 kbar:

```
pt.calculate_ol_liq_temp(liq_comps=myLiquids1, ol_comps=myOls1,
equationT="T_Put2008_Eq21", P=5)
```

By default, this function returns a panda dataframe with the temperature in Kelvin as well as the mea-

sured $K_{D,\ Fe-Mg}^{Ol-Liq}$ using entered olivine and liquid compositions. If users specify "eq_tests=True", the function will also assess the degree of $K_{D,\ Fe-Mg}$ equilibrium between paired liquids and olivines using all three models for equilibrium discussed above: Toplis (calculated Kd), Matzen (Kd=0.34), and Roeder (Kd=0.3). The outputted panda dataframe will contain columns for $\Delta K_{D,\ Fe-Mg}$ calculated using these different equilibrium tests.

	T_K_calc	Kd Meas	Kd calc (Toplis)	ΔKd, Toplis	ΔKd, Roeder	ΔKd, Matzen	SiO2_Liq
0	1306.09	0.31	0.33	0.02	0.01	0.02	57.02
1	1240.35	0.18	0.31	0.14	0.12	0.15	57.66
2	1286.37	0.27	0.32	0.05	0.03	0.06	60.73

6.1 Clinopyroxene-Liquid Thermobarometry

Thermobar contains a number of different thermometers/barometers applicable to clinopyroxene-liquid pairs (Fig. 4). In the most simplic case, where relevant clinopyroxene-liquid pairs have been identified (e.g., experimental produts), data should be formatted as an excel spreadsheet where each row contains a matched pair of liquid and clinopyroxene composition. The function **calculate_cpx_liq_press** allows users to calculate pressures for a variety of barometers, while the function **calculate_cpx_liq_temp** calculates temperature. For thermometers which are P-sensitive users will have to enter a pressure in kbar, or for T-sensitive barometers, a temperature in K. The following code snippet calculates temperatures using equation 33 of Putirka [2008] at 5 kbar:

```
pt.calculate_cpx_liq_temp(liq_comps=myLiquids1,
cpx_comps=myCpxs1, equationT="T_Put2008_eq33", P=5)
```

When neither pressure or temperature is known, the function **calculate_cpx_liq_press_temp** iterates towards a solution using a user-supplied pressure and temperature by specifying an equation for both pressure and temp (as shown for Cpx-only in the preceeding section).

A number of methods have been developed to perform clinopyroxene-liquid thermometry by compared all erupted clinopyroxene and liquid compositions from a given volcanic centre/region, and identifying liquid-cpx pairs which meet certain equilibrium criteria (e.g., Neave and Putirka [2017], Neave et al. [2019], Winpenny and Maclennan [2011], Scruggs and Putirka [2018]). In Thermobar, the function **calculate_cpx_liq_press_temp_matching** assesses all possible clinopyroxene-liquid pairs for a user-supplied dataframe of liquid compositions of length N1 (e.g., all XRF analyses from a given volcanic center), and a user-supplied dataframe of measured clinopyroxene compositions of length N2. The function performs the following steps:

1. Liquid components and clinopyroxene components (e.g., cation fractions) are calculated for each individual clinopyroxene and liquid (saving computational time vs. calculating them after the duplication steps below).

2. Each clinopyroxene composition (raw+components) is duplicated N1 times forming a panda dataframe with rows for cpx1-cpx1-cpx1, ..., cpx2-cpx2-cpx2. The dataframe of liquid compositions (raw+components) is duplicated N2 times forming a dataframe of the form Liq1-Liq2-Liq3...LiqN1, Liq-Liq2-Liq3...LiqN1, These dataframes are combined, creating a dataframe of length N1*N2 with all possible clinopyroxene-liquid pairings of the format Cpx1-Liq1, Cpx1-Liq2, Cpx1-Liq3, Cpx2-Liq1... etc).

3. Compositional components which require both a liquid and cpx composition are calculated for this combined dataframe (e.g., the DiHd component, and $K_{D,\ Fe-Mg}^{Cpx-Liq}$).

As cpx-liquid equilibrium tests are sensitive to pressure and/or temperature, equilibrium tests cannot be performed until pressures and temperatures for each pair have been calculated. However, calculating pressures and temperatures iteratively for all possible clinopyroxene-liquid matches can be very time consuming (e.g., 400 clinopyroxenes and 2500 possible liquids requires 1 million iterative calculations to be performed). To increase computational efficiency, we apply a preliminary filter in terms of $K_{D,\ Fe-Mg}^{Cpx-Liq}$ equilibrium (using equation 35 of Putirka [2008] by default). As $K_{D,\ Fe-Mg}^{Cpx-Liq}$ parametrizations are not pressure-sensitive, so we use the **calculate_cpx_liq_temp** function to calculate a minimum temperature for each clinopyroxene (for P=-10 bars), and a maximum temperature (for a default maximum pressure of 30 kbars). This upper pressure limit was set with volcanic systems in mind, but can be easily overridden when calling the function by setting PMax=" ". These maximum and minimum equilibrium $K_{D,\ Fe-Mg}^{Cpx-Liq}$ values are compared to the measured $K_{D,\ Fe-Mg}^{Cpx-Liq}$ values for each cpx-liquid pairs. If the deviation between measured and calculated $K_{D,\ Fe-Mg}$ is greater than 0.03 (the default value, changed by specifying KdErr=" ") for both the minimum and maximum equilibrium $K_{D,\ Fe-Mg}$, no temperatures in-between will yield a match. Thus, cpx-liquid matches which lie outside this critical equilibrium value can be discarded.

5. The function **calculate_cpx_liq_press_temp** is used to iteratively calculate pressures and tem-

peratures for remaining clinopyroxene-liquid pairs.

6. Using the calculated temperature and pressure for each pair, the equilibrium $K_{D,\ Fe–Mg}$ is calculated using equation 35 of Putirka [2008] (although this default can be overridden), the equilibrium CaTs component using the expression of Putirka [1999], and the updated equilibrium EnFs and DiHd components calculated using the expression of Mollo et al. [2013], following Neave et al. [2019]. It is worth noting that in supplementary spreadsheet of Neave et al. [2019] uses the Putirka (1996) anhydrous thermometer to calculate the $K_{D,\ Fe–Mg}$ component, while temperature is calculated using equation 33. In our code, $K_{D,\ Fe–Mg}$ is calculated using the user-specified thermometer for consistency.

7. By default, the code then selects cpx-liquid pairs where the measured components (calculated using the method of Putirka et al. [1996]) and calculated equilibrium components are within 0.03 for $K_{D,\ Fe–Mg}$, 0.06 for DiHd, 0.05 for EnFs, and 0.03 for CaTs (following the supporting excel spreadsheet of Neave et al. [2019]). Users can changes these selection criteria using the criteria "sigma=" and "KdErr=". For example, if sigma=2, the values for DiHd, EnFs and CaTs are doubled. If KdErr=0.06, matches within ± 0.06 are considered.

8. Following the approach of Neave and Putirka [2017], the code also performs calculations to average the pressures and temperatures for each cpx. For example, if Cpx1 matches with Liq1, Liq3, and Liq9, the values for these three matches will be averaged, and the standard deviation of the pressure and temperature are returned.

9. The function returns a dictionary. Users can extract a panda dataframe of all liquid-cpx matches which meet the specified equilibrium criteria using dictionary['All_PTs']. The second part of the dictionary (accessed using dictionary['Av_PTs'] contains the average and standard deviation for each clinopyroxene.

The speed at which these calculations are performed are significantly faster than previous tools (seconds vs. tens of minutes for ~300,000 clinopyroxene-liquid pairs). This, along with the flexibility provided by the implementation of these tools in python, offers users a lot more freedom to assess possible melt-clinopyroxene matches in larger datasets. For example, there are a number of inputs that users can customize when performing clinopyroxene-melt matching:

- **KdMatch**. This overrides the default, which calculates the ideal value of $K_{D,\ Fe–Mg}$ for a liquid-cpx pair is calculated using equation 35 of Putirka (2008) as a function of temperature. For example, users could specify KdMatch=0.27, which would compare measured $K_{D,\ Fe–Mg}$ values to a hypothetical value of 0.27. Or, users can specify KdMatch="Masotta", which calculates $K_{D,\ Fe–Mg}$ using equation 35Alk of Masotta et al. [2013]. This equation expresses $K_{D,\ Fe–Mg}$ as a function of temperature, and the cation fractions of Na_2O and K_2O in the melt, and was developed for trachyte and phonolitic magmas (extreme care should be taken when applying it to other melt compositions).

- **KdErr**: This allows users to overwrite the default filter where only cpx-liquid matches with measured and predicted $K_{D,\ Fe–Mg}$ values within 0.03 are used to calculate P and T. For example, users could specify KdErr=0.08, to consider all pairs within 0.08 units of equilibrium.

- **sigma**. By default, the code uses 1 sigma values from Neave et al. (2019) for EnFs (0.05), DiHd (0.06) and CaTs (0.03) equilibrium tests. If users were to specify sigma=2, pairs within 0.01, 0.12 and 0.06 would be considered.

- **Eq_Crit**. This allows users to select which equilibrium tests they want to use. Default is "All", which uses Kd, DiHd, EnFs, CaTs. If Eq_Crit ="Kd", pairs are only filtered based on Kd. If Eq_Crit ="Kd_DiHd", only filters based on $K_{D,\ Fe–Mg}$ and DiHd. If Eq_Crit ="Kd_EnFs", only filters based on $K_{D,\ Fe–Mg}$ and EnFs.

- **Cpx_Quality**. By default, all clinopyroxenes are considered. If users specify "Cpx_Quality"=True, only cpxs passing the analysis quality tests of Neave et al. (2019) are considered (Cation sum between 3.99 and 4.02, and cpx Jd component >0.01).

- **Fe3FeT_Liq, H2O_Liq**. Rather than having to alter the input dataframe of liquids (as shown in Fig. 8), users can specify a new H2O content and Fe3FeT ratio in the function itself. This can be a fixed value for all calculations, or could be set as a panda series with the same length as the input dataframe of liquid compositions.

6.1.1 *Recreating Scruggs and Putirka (2008)*

To demonstrate the versatility of this cpx-liq melt matching function, we recreate the analysis of Scruggs and Putirka [2018], who perform clinopyroxene-liquid equilibrium on samples from Chaos Craggs at Lassen Peak. The erupted liquids sampled at Chaos Craggs are strongly bimodal. To capture the compositions of liquids which likely exist at depth plotting between these two erupted end-members, Scruggs and Putirka [2018] add or subtract the composition of a felsic-whole rock composition from measured mafic liquids, and use the solver functions in excel to find the mixing proportion that best satisfies equilibrium tests (https://www.youtube.com/watch?v=CjKvgXrah_k&list=PLnOXMT9X-AL_No_vUkkx8tYrahGQ1X4Kh&index=2&t=13s).

We demonstrate how a similar, but more automated approach can be implemented in Thermobar in Fig. 8, and this worked example is also provided as a Jupyter Notebook in the supporting information. Step 1 imports an excel spreadsheet containing possible liquid compositions (whole-rock data in this example), and a separate sheet or spreadsheet containing measured clinopyroxene compositions (Fig. 7). Step 2 uses the function **add_noise_sample_1phase** to make a silicic end-member to use for mixing. We apply a filter to only consider liquids with > 65 wt% SiO_2, and for each measured liquid, we generate 5 duplicates, adding normally-distributed noise with 1σ=1% to all oxides. This helps to account for the fact that there are also a number of silicic liquids which exist at depth, but are not represented in sampling. We use the same function to make synthetic liquids based on the composition of measured samples with < 53.8 wt% SiO_2 and > 4 wt% MgO for the mafic end member. The following steps could also be performed using a dataframe of liquid compositions without any noise or filters added.

Step 3 mixes these end-members to generate synthetic liquids spanning the entire compositional range between measured liquids. Thermobar provides a number of options within the function **calculate_bootstrap_mixes** to mix two end members in various proportions (all of which are discussed in the example notebook). In its simplest form, this function takes two end members, and mixes a randomly-selected composition from one end member with a randomly-selected composition from the other end member, with the mixing proportion varying randomly between 0 and 1. Additional flexibility is provided by the optional input "self_mixing". If "self_mixing=True", the two end members are combined into a single dataframe, and these compositions are randomly mixed. This means that mixing happens not only between mafic and silicic end members (as in the default form), but also mafic end member compositions are mixed with other mafic end-member compositions, and silicic end-member compositions are mixed with other silicic end-member compositions. This method produces a strong clustering of synthetic liquids near the end members, which may be useful in certain circumstances. However, in this specific example, relatively few liquids generated by this function lie within the compositional gap between mafic and silicic compositions for <1000 duplicates. Thus, we use the option self_mixing="Partial", which creates half the mixes by mixing between silicic and mafic end members, and the other half from self-mixing.

Step 4 is optional (Fig. 8), and combines this synthetic dataframe of liquids with the original dataframe of liquids using the pandas concat function (to include samples which weren't selected as end members). Finally, because clinopyroxene thermometry is sensitive to the H_2O of the liquid, but H_2O contents at depth cannot be deduced from bulk rock analyses of degassed lava samples, Scruggs and Putirka [2018] calculate the H_2O of the liquid as a function of the SiO_2 content. Here, setting Combined_Liqs['H2O_Liq']= overwrites any H_2O contents entered in the original liquid input.

Step 5 (Fig. 8) inputs this finalized dataframe of generated liquids and measured clinopyroxene compositions into the function **calculate_cpx_liq_press_temp_matching**. The H_2O content of the liquid using the expression of Scruggs and Putirka [2018] could also have been entered at this stage using the input "H2O_Liq". Step 6 uses matplotlib to plot averaged pressures and temperatures from each clinopyroxene as red diamonds with 1σ error bars (plt.errorbar), and all possible matches as semi-transparent symbols.

Step 1 - Import all measured Liquids and Cpxs

```
out=pt.import_excel('Scruggs_Input.xlsx', sheet_name="Liquids")
my_input=out['my_input']
myLiquids1=out['Liqs']

out2=pt.import_excel('Scruggs_Input.xlsx', sheet_name="Cpxs")
my_input2=out2['my_input']
myCpxs1=out2['Cpxs']
```

Extracts df of liquid compositions

Extracts df of cpx compositions

Step 2 - Generate Silicic and Mafic end-members (adding noise)

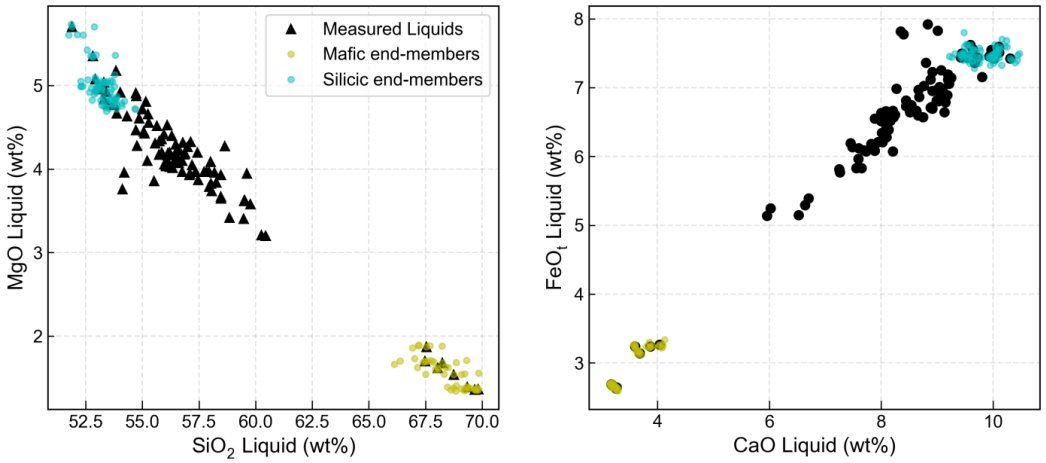
```
Sil_endmember_noise1=pt.add_noise_sample_1phase(phase_comp=myLiquids1,  duplicates=5, filter_q='SiO2_Liq > 65',
                    phase_err_type="Perc", noise_percent=1, err_dist="normal", append=True)

Maf_endmember_noise1=pt.add_noise_sample_1phase(phase_comp=myLiquids1, duplicates=5, filter_q='SiO2_Liq < 53.8 & MgO_Liq>4',
                    phase_err_type="Perc", noise_percent=1, err_dist="normal", append=True)
```

Creates 5 duplicates per sample

Compositional filter

Appends liqs passing compositional filter to new noisy samples

Add normally distributed noise (1σ=1%)



Step 3 - Generate synthetic liquids by mixing end-members

```
Mixed_noise1_selfbig=pt.calculate_bootstrap_mixes(endmember1=Sil_endmember_noise1,
                    endmember2=Maf_endmember_noise1, num_samples = 500, self_mixing = "Partial")
```

50% mixes between end members
50% mixes within each end member

Generates 500 synthetic liquids

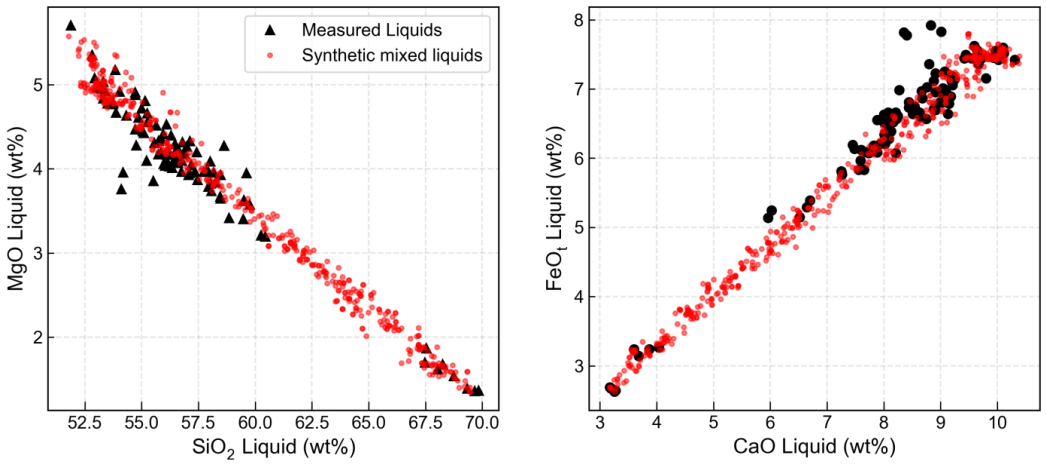


Figure 7: Example of functions allowing users to generate synthetic liquids, adapted from the approach of Scruggs and Putirka [2018]. Step 1: The user reads in all measured clinopyroxene compositions into one panda dataframe (myCpxs1), and all liquids into a second dataframe (MyLiquids1). Step 2: Using as many "filters" as required, the user defines 2 end members. These end members are then mixed to generate 500 synthetic liquids which incorperate the variation in the natural data.

Step 4 - Combine synthetic liquids and measured liquids ¶

```
Liq_Comp=pd.concat([Mixed_noise1_selfbig.reset_index(drop=True),
        myLiquids1.reset_index(drop=True)], axis=0).reset_index(drop=True).fillna(0)
```

Gets rid of indexing issues, removes Nans

Step 5 - Set water content (following Scruggs and Putirka, 2018)

```
Liq_Comp['H2O_Liq']=Liq_Comp['SiO2_Liq']*0.06995+0.383
```

Overwrites water contents with a value dependent on SiO2

Step 6 - Perform melt matching to calculate pressures and temperatures

```
melt_match_out_syn=pt.calculate_cpx_liq_press_temp_matching(liq_comps=Liq_Comp, cpx_comps=myCpxs1,
                    equationP="P_Neave2017", equationT="T_Put2008_eq33", KdMatch=0.27, KdErr=0.03)
Syn_Avs=melt_match_out_syn['Av_PTs']
Syn_All=melt_match_out_syn['All_PTs']
```

Average P and T per cpx

All Matches

Considers matches with Kd=0.27±0.03

Syn Average dataframe:

	No. of liquids averaged	Sample_ID_Cpx	Mean_T_K_calc	st_dev_T_K_calc	Mean_P_kbar_calc	st_dev_P_kbar_calc	Mean_Delta_Kd_Put2008	Mean_Delta_Kd_Mas2013	Mean_Delta_EnFs	Mean_Delta_CaTs
0	112	12	1306.874096	8.462061	-0.533247	0.411903	0.044293	0.144914	0.042185	0.016345
0	56	16	1281.141557	18.491046	1.390111	0.746137	0.056823	0.149984	0.004690	0.010030
0	5	26	1304.251537	4.271816	0.623434	0.441238	0.024975	0.131124	0.040744	0.028638
0	217	29	1308.642525	11.814991	-0.498931	0.506761	0.016348	0.119370	0.039149	0.010773
0	11	30	1302.787795	7.103075	-0.561921	0.219530	0.047838	0.146350	0.040597	0.021698
...	...	...	...	...	...	...	...	...	...	...

Plotting average for each Cpx, with all matches underlain

```
fig, (ax1) = plt.subplots(1, 1, figsize=(6, 5))
ax1.plot(Syn_All['T_K_calc'],  Syn_All['P_kbar_calc'], 'or',
         ms=2, mfc='red', alpha=0.1, label='All Matches')
ax1.errorbar(Syn_Avs['Mean_T_K_calc'],  Syn_Avs['Mean_P_kbar_calc'],
             xerr=Syn_Avs['st_dev_T_K_calc'],
             yerr=Syn_Avs['st_dev_P_kbar_calc'],
             fmt='d', ecolor='k', elinewidth=0.8,
             mfc='red', ms=8, mec='k', label='Average Matches')
ax1.invert_yaxis()
ax1.set_xlabel('Temp (K)')
ax1.set_ylabel('Pressure (kbar)')
ax1.legend()
fig.savefig('AllMatches_PT.png', dpi=300)
```

Plots light red circles, all cpx-liq matches

Plots error bar for the average PT for each Cpx

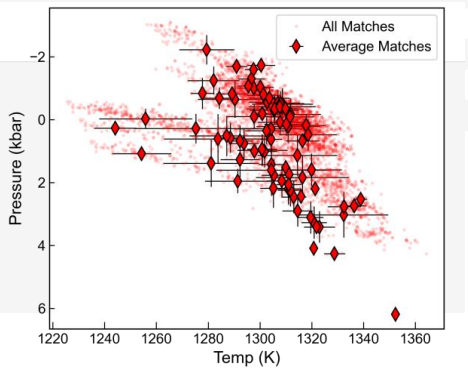


Figure 8: Once synthetic liquids have been calculated, users may wish to combine them with measured liquid compositions to get the largest number of available comparisons (step 5). Columns in this combined dataframe can be easily overwritten - Here, the liquid H_2O content is calculated from the SiO_2 content of the liquid (following Scruggs and Putirka [2018]). Once the liquid input is set, the function calculate_cpx_liq_press_temp_matching is called, specifying the choice of liquid and clinopyroxene compositions, as well as the equation for pressure and temperature. By default, this function filters clinopyroxene-liquid pairs using the criteria in the supporting information of Neave et al. [2019]. This function returns a dictionary, which can be subdivided into a panda dataframe containing all matches, and a dataframe where pressures and temperatures have been averaged for all the liquids in equilibrium with a given clinopyroxene composition. Plotting both outputs gives insight into the amount of scatter associated with each liq-cpx pair compared to averaged outputs.

6.2 Orthopyroxene-Liquid Thermobarometry

The orthopyroxene-liquid functions in Thermobar are very similar to those for cpx-liquid. If users wish to calculate pressure or temperature for known liquid-opx matches (e.g., measured rim and matrix glass compositions), they can use the functions **calculate_opx_liq_press** and **calculate_opx_liq_temp**, specifying the compositions of liquids and orthopyroxenes as in previous examples, and selecting an equation for pressure or temperature from figure 4. Similarly, P and T can be solved iteratively using **calculate_opx_liq_press_temp**, specifying an equationP and equationT.

As for cpx-liquid, the function **calculate_opx_liq_press_temp_matching** assesses all possible liquid and orthopyroxene pairs, and calculates P and T for those within user-specified ranges for equilibrium. Unlike for clinopyroxene-liquid, there is only one commonly used equilbrium test for orthopyroxene-liquid pairs, which compares measured values of $K_{D,\ Fe-Mg}^{Opx-Liq}$ to those predicted from the liquid composition. Putirka [2008] suggest that the range of $K_{D,\ Fe-Mg}$ values in experiments ranges from 0.29±0.06, and can also be expressed as a function of the cation fraction of Si in the liquid ($K_{D,\ Fe-Mg}$=0.4805-0.3773 X_{Si}^{liq}). Because this value is independent of P and T, this filter can be applied before iterating (which simplifies the function relative to that for cpx-liq). The opx-liquid melt matching algorithm follows steps 1-3 described in Section 6.1. Then, $K_{D,\ Fe-Mg}^{Opx-Liq}$ values are computed for each opx-liquid pair, and compared to equilibrium values. By default, the function calculates equilibrium values using the X_{Si}^{liq} expression of Putirka [2008], and considers all matches within $\Delta K_{D,\ Fe-Mg}$ of 0.06. Users can override this default option by specifying a value for KdMatch, and KdErr in the function. For example, in the code snippet below, orthopyroxene-liquid pairs with measured $K_{D,\ Fe-Mg}^{Opx-Liq}$ within 0.29±0.07 are evaluated:

```
PxLi_PT=calculate_Opx_Liq_PT_melt_matching(Liq_Comps=myLiquids1,
                Opx_Comps=MyOpxs, KdMatch=0.29, KdErr=0.07)
```

Following this filtering step, the function takes pairs in equilibrium and uses the **calculate_opx_liq_press_temp** function to calculate pressure and temperature for each pair. A dictionary is returned, containing the pressure and temperature for each pair (accessed using dictionary['All_PTs']. As for clinopyroxene-liquid, a second output is also calculated, where all matches for a given orthopyroxene are averaged (e.g., Opx1-Liq1, Opx1-Liq10, Opx1-Liq32), accessed using dictionary['Av_PTs']. Users also have the option to overwrite the Fe3FeT_Liq value specified in input, as this function uses only Fe^{2+} species to calculate $K_{D,\ Fe-Mg}$.

6.3 Two pyroxene Thermobarometry

As for clinopyroxene-liquid, and orthopyroxene-liquid, the function **calculate_cpx_opx_temp** allows users to calculate temperatures for matched clinopyroxene-orthopyroxene pairs, **calculate_cpx_opx_Press** calculates pressures, and **calculate_cpx_opx_press_temp** iterates towards a pressure and temperature if an equation for pressure and temperature are selected.

calculate_cpx_opx_press _temp_matching assesses all possible clinopyroxene-orthopyroxene pairs. As for orthopyroxene-liquid, the partitioning of Fe-Mg between orthopyroxene-clinopyroxene is the only available equilibrium test. Because of the variation in this parameter between different systems, by default, the code returns all possible clinopyroxene-orthopyroxene pairs. If users specify KdMatch="HighTemp", the code will calculate pressures and temperatures for all cpx-opx pairs with $K_{D,\ Fe-Mg}^{Cpx-Opx}$=1.09 ± 0.14 suggested by Putirka [2008] for high temperature systems. If users specify KdMatch="LowTemp", all matches within 0.7 ± 0.2 are used (following the suggestions for subsolidus systems of Putirka [2008]). As for clinopyroxene- and orthopyroxene-liquid, users can put a value for KdMatch and specify a different value of KdErr (accepting matches within KdMatch±KdErr). As for clinopyroxene- and orthopyroxene-liquid matching, the function returns a dictionary containing pressures and temperatures for all matches, as well as pressures and temperatures averaged for each clinopyroxene.

6.4 Plagioclase-Liquid and Alkali Feldspar-Liquid Thermobarometery

Plagioclase-Liquid and alkali feldspar-liquid thermobarometry are considered in generic functions **calculate_fspar_liq_temp**, **calculate_fspar_liq_press**, and **calculate_fspar_liq_press_temp**, because the mineral component calculations of Putirka [2008] are the same for all feldspar end-members. If users are inputting plagioclase compositions, they should specify plag_comps=" " in the function, and for alkali feldspars kspar_comps=" ".

Equilibrium tests are currently only implemented for plagioclase, comparing the calculated and predicted An, Ab and Or components between plagioclase and liquid. In particular, Putirka (2008) suggest that the Ab-An exchange coefficient is a good equilibrium test, as it varies little as a function of pressure, temperature or melt H_2O content (~0.27±0.18). In their supporting spreadsheet updated since 2008, they suggest using val-

ues of 0.28±0.11 for T>1050°C, and 0.1±0.05 for T<1050°C. In the example jupyter notebook, we demonstrate how to filter pairs to only take those passing this equilibrium criteria.

6.5 Plagioclase Hygrometers

The function **calculate_fspar_liq_hygr** allows the H_2O contents of liquids which crystallized plagioclase to be estimated. These hygrometers require users to specify the composition of the liquid, as well as the anorthite and albite content of each plagioclase. Analogous to the other functions, the composition of liquids and plagioclase dataframes are specified in the function, along with the pressure and temperature:

```
calc=pt.calculate_fspar_liq_hygr(liq_comps=myLiquids1,
plag_comps=myPlags1, equationH="H_Waters2015", T=1300, P=5)
```

	Pass An-Ab Eq Test Put2008?	H2O_calc	Delta_An	Delta_Ab	Delta_Or	Pred_An_EqE
0	Low T: Yes	2.183611	0.056252	0.141146	0.029165	0.360876
1	Low T: Yes	2.671574	0.083157	0.227579	0.028164	0.369968

This returns a pandas dataframe of the calculated H_2O content, along with an indicator of whether the pair passed the recommended equilibrium test of Putirka (2008) based on the temperature inputted by the user.

Alternatively, users can just enter the anorthite and albite content of the plagioclase, without requiring the full plagioclase composition:

```
pt.calculate_fspar_liq_hygr(liq_comps=Liqs_PL, XAn=0.5, XAb=0.4,
    equationH="H_Waters2015", T=1300, P=5)
```

In the example notebook, we also show how liquid-plagioclase thermometers and hygrometers can be iterated when neither temperature nor H_2O content is known (as the biggest limitation of plagioclase-liquid hygrometers is that they are extremely sensitive temperature).

6.6 Two feldspar Thermobarometry

Temperature from co-existing kspar-plag pairs can be calculated using the function **calculate_plag_kspar_temp**. Analogous to the function for clinopyroxene-liquid, orthopyroxene-liquid and orthopyroxene-clinopyroxene, the function **calculate_plag_kspar_temp_matching** considers all possible pairs between a dataframe of plagioclase compositions, and a dataframe of k-feldspar compositions. Putirka (2008) suggest that a comparison of activities for An, Ab and Or in plagioclase and k-feldspar using the models of Elkins and Grove [1990] can be used as an equilibrium test, although they note that while the values should nominally by zero, further examination of experimental data is required to determine reasonable cut offs analagous to those used for cpx-liq and opx-liq. Thus, Thermobar returns these values for matching pairs if the user specifies "eq_tests=True", and returns them automatically for the matching function. The example jupyter notebook shows users how they could filter pairs using different values or these equilibrium tests.

7 Error propagation

Estimating uncertainty when performing thermobarometry and hygrometery calculations is important, as many calibrations are highly sensitive to the concentration of minor components which are difficult to measure with high precision (e.g., Na_2O and Al_2O in clinopyroxene). Additionally, sometimes parameters like melt H_2O contents are poorly known, particularly for volcanic systems where melt inclusion analyses are sparse, or non existant.

The function **add_noise_sample_1phase** can be used to generate duplicates of rows in a user-inputted dataframe with a specified amount of noise added. There are a number of ways of how to use this function (e.g., adding uniform vs. normally distributed, percentage or absolute errors), which are discussed in detail in the example Jupyter Notebook and the documentation. Figure 10 shows a worked example calculating the distribution of pressures for the average reported clinopyroxene-liquid composition in each experiment from Feig et al. [2010] resulting from a normally distributed error of 5% in the Na_2O component of clinopyroxene. Ignoring all other sources of error, this optimistic estimate of the error associated with EPMA analyses of just a single oxide generates a pressure distribution with 1σ=0.2-0.3 kbar.

Figure 9 shows a worked example where oxide concentrations in both the melt and clinopyroxene are generated for a normal distribution using the published 1σ uncertainties for each experiment. The resulting pressure distributions have 1σ values ranging from 0.4-2.2 kbar. This demonstrates that a combination of analytical uncertainty and experimental noise is one of the reasons that experimental pressures and pressures calculated using mineral-melt barometers rarely show a strong correlation for any given experimental study.

These tools allow users to estimate the uncertainty resulting from their specific analytical conditions. For example, repeated measurements of secondary mineral standard with similar elemental concentrations could be used to calculate the minimum error on barometry, thermometry and hygrometry calculations.

Step 1 - Import Cpx and Liquid data

```
out=pt.import_excel('Cpx_Liq_error_prop_Feig2010_example.xlsx', sheet_name="Sheet1")
my_input=out['my_input']
myCpxs1=out['Cpxs']
myLiquids1=out['Liqs']
```

Extracts dataframes of liquid and cpx compositions

Step 2 - Add 5% error to Na2O in Cpx, no error to liquid compositions

```
Cpx_5Na2O=pt.add_noise_sample_1phase(phase_comp=myCpxs1, variable="Na2O",
          variable_err=5, variable_err_type="Perc", duplicates=1000,
          err_dist="normal")

Liquids_only_noNoise=pt.add_noise_sample_1phase(phase_comp=myLiquids1,
                      noise_percent=0, duplicates=1000, err_dist="normal")
```

Adds normally distributed noise with 1σ=5% to Na_2O. Makes 1000 duplicates per user-entered row

Duplicates liquids to have the same shape as cpxs but without adding any error

Step 3 - Calculate pressures and temperatures iteratively

```
Out_5_noise_cpx=pt.calculate_cpx_liq_press_temp(liq_comps=Liquids_only_noNoise, cpx_comps=Cpx_5Na2O,
                                  equationP="P_Put2008_eq31", equationT="T_Put2008_eq33", eq_tests=True)
```

Step 4 - Calculate Statistics for each inputted cpx-liq pair

```
Stats_P_kbar=pt.av_noise_samples_series(Out_5_noise_cpx['P_kbar_calc'], Out_5_noise_cpx['Sample_ID_Liq_Num'])
Stats_P_kbar
```

	Sample	Mean_calc	Median_calc	St_dev_calc	Max_calc	Min_calc
0	0.0	3.945895	3.941735	0.253565	4.758968	2.989067
1	1.0	3.743047	3.750585	0.229589	4.493188	2.948342
2	2.0	4.347602	4.355944	0.252757	5.150464	3.365308

Step 5 - Plot histogram for the first entered cpx-liq pair

```
fig, ((ax1)) = plt.subplots(1, 1, figsize=(7, 5))
ax1.hist(Out_5_noise_cpx.loc[Out_5_noise_cpx['Sample_ID_Cpx_Num']==0,
                             "P_kbar_calc"], bins=50, density = True)
ax1.annotate("Liq1-Cpx1", xy=(0.02, 0.95), xycoords="axes fraction",
             fontsize=12)
ax1.set_xlabel('Pressure (kbar)')
ax1.set_ylabel('Probability Density')
fig.savefig('5%error_cpx.png', dpi=300)
```

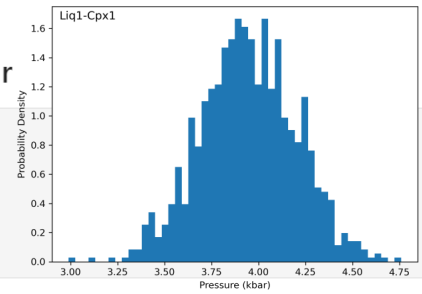


Figure 9: Example demonstrating how to propagate error in Na_2O in clinopyroxene into pressure. Step 1 reads in data from an excel spreadsheet (containing oxide concentrations). Step 2 generates 1000 duplicates per row loaded in step 1, where these duplicates have normally distributed error with 1σ=5% added to just the Na_2O component of clinopyroxene. A dataframe of liquid compositions is also generated which is the same size as the clinopyroxene dataframe, but has no added noise. Step 3 calculates pressures and temperatures iteratively. Step 4 averages the calculated pressures and temperatures for all rows with the same sample name, and prints statistics for these averages (e.g, mean, median, max calculated pressure, min calculated pressure, standard deviation). Step 5 plots a histogram of calculated pressures for the cpx-liquid pair in the first row of the spreadsheet.

Input spreadsheet (columns for absolute error in wt%)

	A	B	C	D	E	F	G	H	I
	Label	SiO2_Liq	SiO2_Liq_Err	TiO2_Liq	TiO2_Liq_Err	Al2O3_Liq	Al2O3_Liq_Err	FeOt_Liq	FeOt_Liq_Err
	Feig2010_pair1	50.97	0.33	0.49	0.04	19.35	0.22	5.33	0.43
	Feig2010_pair2	53.64	0.33	0.62	0.03	19.32	0.24	4.88	0.21
	Feig2010_pair3	49.63	0.48	0.37	0.03	19.10	0.24	5.30	0.30

Step 1 - Import Cpx and Liquid data

```
out=pt.import_excel('Cpx_Liq_error_prop_Feig2010_example.xlsx', sheet_name="Sheet1")
my_input=out['my_input']
myCpxs1=out['Cpxs']
myLiquids1=out['Liqs']
```

Extracts dataframes of liquid and cpx compositions

Step 2 - Import Errors for cpx and Liquids

```
out_err=pt.import_excel_errors('Cpx_Liq_error_prop_Feig2010_example.xlsx', sheet_name="Sheet1")
myLiquids1_err=out_err['Liqs_Err']
myCpxs1_err=out_err['Cpxs_Err']
myinput_Out=out_err['my_input_Err']
```

Extracts dataframes of errors from columns with _Err

Step 3 - create 1000 duplicates per row with normally-distributed noise based on published 1σ

Specifies dataframe with errors in

```
Liquids_st_noise=pt.add_noise_sample_1phase(phase_comp=myLiquids1, phase_err=myLiquids1_err,
                                            phase_err_type="Abs", duplicates=1000, err_dist="normal")

Cpxs_st_noise=pt.add_noise_sample_1phase(phase_comp=myCpxs1, phase_err=myCpxs1_err,
                                         phase_err_type="Abs", duplicates=1000, err_dist="normal")
```

Specifies errors should be normally distributed

Specifies 1000 duplicates per row

Step 4 - Calculate pressures and temperatures iteratively for these new dataframes

Specifies 2 dataframes with added noise

```
Out_st_noise=pt.calculate_cpx_liq_press_temp(liq_comps=Liquids_st_noise, cpx_comps=Cpxs_st_noise,
            equationP="P_Put2008_eq31", equationT="T_Put2008_eq33", eq_tests=True)
```

Also calculates equilibrium tests

Specifies which barometer and thermometer to iterate

Step 5 - Calculate statistics for each inputted cpx-liq pair

Specifies identifier to average by (this column is added in step 3 to aid averaging)

```
Stats_P_kbar=pt.av_noise_samples_series(Out_st_noise['P_kbar_calc'], Liquids_st_noise['Sample_ID_Liq_Num'])
Stats_P_kbar # prints this line
```

Specifies column to perform statistics on

	Sample	Mean_calc	Median_calc	St_dev_calc	Max_calc	Min_calc
0	0.0	3.889951	3.940285	0.839779	6.077707	0.137175
1	1.0	3.447472	3.724100	1.756871	6.548999	-17.761073
2	2.0	3.942508	4.174495	2.110455	8.883845	-6.526874

Each row is the statistics for 1000 synthetic liquids and cpxs for the original synthetic dataset

Step 6 - Histograms of calculated pressures for 2 input pairs

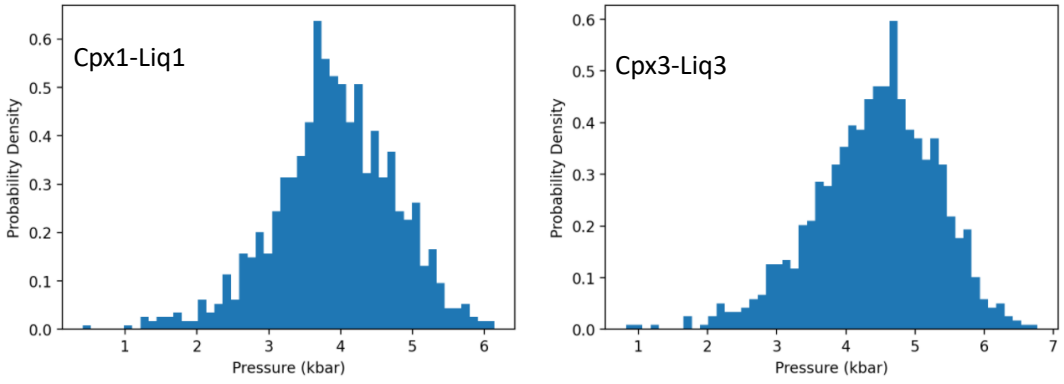


Figure 10: Example of functions allowing users to assess how error in oxide concentrations propagates into calculated pressures. Step 1 and 2 reads in data from an excel spreadsheet (containing oxide concentrations and absolute errors). Step 3 specifies the type of error to be added; in this case, 1000 synthetic liquids and cpxs are made for each user-inputed row, and error is added based on the errors stored in the dataframes created in step 2 (normally distributed error with a standard deviation equal to the published 1σ error value). Step 4 inputs these dataframes into the function to iterate pressure and temperature. Step 5 averages the calculated pressures and temperatures for all rows with the same sample name, and prints statistics for these averages (e.g, mean, median, max calculated presure, min calculate pressure, standard deviation). The distribution of pressures for each cpx-liq pair can also been shown on histograms (see Jupyter Notebook for plotting code)

8 Integration with other open-source Python tools

In the last few years, there has been an increase in the number of petrological tools available in python (e.g., Pyrolite for geochemical plotting: Williams et al. [2020], MiMIC for melt inclusion modification: Rasmussen et al. [2020], VESIcal for volatile solubility Iacovino et al. [2021]. Having thermobarmotery tools available in python through Thermobar will allow increased integration between various codes. For example, one of the most common uses of volatile solubility models is to calculate the pressure at which a melt inclusion was trapped based on reconstructing its H_2O, CO_2, and major element contents at the time of melt inclusion entrapment. To convert these chemical parameters into a pressure, the temperature of the melt inclusion at the time of entrapment must also be estimated. In the supporting information, we show how the functions **convert_to_VESIcal** and **convert_From_VESIcal** can be used to convert oxide data back and forth from the formats used in Thermobar and VESIcal so the tools can be used together.

9 Future Work

The open-source nature of Thermobar, with code available on github, means that users can adapt functions, add their own, or incorporate new thermobarometery or hygrometry equations as they are published. Authors publihing new thermobarometry equations can contact the author team of Thermobar, and an effort will be made to continue to update the available equations. To reflect the probable evolving nature of this tool, when citing Thermobar, users should specify which version they used, as well as citing the original equations used for calculations. For example "cpx-liquid pressures and temperatures were calculated using equation 30 and 31 of Putirka (2008), implemented through the python3 tool Thermobar (V.1.0.1, Wieser et al. 2021). Ideally, users should provide the jupyter notebook used for calculates for maximum repeatability, and to outline the various options used (particularly for more complicated operation such as melt matching, error propagation.

10 Conclusions

Thermobar provides access to more than 100 popular thermometers, barometers and hygrometers through easy-to-implement and customize functions within the open-source programming language, Python3. In addition to simpler calculations of pressure and temperature, this tool also provides ways to assess all possible equilibrium pairs for a variety of phases, as well as propagating errors in a Monte-Carlo method.

Acknowledgements

We are very grateful to Keith Putirka for answering lots of questions about the implementation of different barometers in his excel spreadsheets, as well as very helpful discussions regarding $K_{D,\ Fe\text{–}Mg}$. We thank Euan Mutch for sharing a spreadsheet for his amphibole barometer, and David Neave for helpful discussions regarding his melt-matching tool. PW thanks Kayla Iacovino and Simon Matthews for introducing her to the wonderful world of developing Open-source python tools.

Author contributions

PW wrote the manuscript and the majority of the python code, as well as performing the benchmarking of this code to existing tools. MP helped with code writing (e.g., bootstrapped liquids), as did GL (e.g., amphibole site occupancy). EW helped optimize computational speed for various iterative calculations, as well as providing guidance for writing documentation in sphinyx, creating a binder file, and making the code available through pip. AK and CT helped conceive the project. All authors provided feedback on the manuscript.

Data availability

All files are avaiable on github ** (rest is future tense) where the code can be run through binder. YouTube videos explaining various aspects of the tool are available on the Thermobar channel https://www.youtube.com/channel/UC7ddceuNnikCdQa_fRHmdXw.

References

Anderson, J. L. and Smith, D. R. (1995). The effects of temperature and fo on the al-in-hornblende barometer. *American Mineralogist*, 80(5-6):549–559.

Balta, J. B., Sanborn, M., McSween Jr, H. Y., and Wadhwa, M. (2013). Magmatic history and parental melt composition of olivine-phyric shergottite lar 06319: Importance of magmatic degassing and olivine antecrysts in martian magmatism. *Meteoritics & Planetary Science*, 48(8):1359–1382.

Beattie, P. (1993). Olivine-melt and orthopyroxene-melt equilibria. *Contributions to Mineralogy and Petrology*, 115(1):103–111.

Blundy, J. D. and Holland, T. J. (1990). Calcic amphibole equilibria and a new amphibole-plagioclase geothermometer. *Contributions to mineralogy and petrology*, 104(2):208–224.

BREY, G. P. and Köhler, T. (1990). Geothermobarometry in four-phase lherzolites ii. new thermobarometers, and practical assessment of existing thermobarometers. *Journal of Petrology*, 31(6):1353–1378.

Brugman, K. K. and Till, C. B. (2019). A low-aluminum clinopyroxene-liquid geothermometer for high-silica magmatic systems. *American Mineralogist: Journal of Earth and Planetary Materials*, 104(7):996–1004.

Coogan, L., Saunders, A., and Wilson, R. (2014). Aluminum-in-olivine thermometry of primitive basalts: Evidence of an anomalously hot mantle source for large igneous provinces. *Chemical Geology*, 368:1–10.

Culha, C., Suckale, J., Keller, T., and Qin, Z. (2020). Crystal fractionation by crystal-driven convection. *Geophysical Research Letters*, 47(4):e2019GL086784.

Edmonds, M., Cashman, K. V., Holness, M., and Jackson, M. (2019). Architecture and dynamics of magma reservoirs.

Elkins, L. T. and Grove, T. L. (1990). Ternary feldspar experiments and thermodynamic models. *American Mineralogist*, 75(5-6):544–559.

Feig, S. T., Koepke, J., and Snow, J. E. (2010). Effect of oxygen fugacity and water on phase equilibria of a hydrous tholeiitic basalt. *Contributions to Mineralogy and Petrology*, 160(4):551–568.

Gaetani, G. A., O'Leary, J. A., Shimizu, N., Bucholz, C. E., and Newville, M. (2012). Rapid reequilibration of h_2o and oxygen fugacity in olivine-hosted melt inclusions. *Geology*, 40(10):915–918.

Gleeson, M. L., Gibson, S. A., and Stock, M. J. (2020). Upper mantle mush zones beneath low melt flux ocean island volcanoes: insights from isla floreana, galápagos. *Journal of Petrology*, 61(11-12):egaa094.

Hammarstrom, J. M. and Zen, E.-a. (1986). Aluminum in hornblende: an empirical igneous geobarometer. *American mineralogist*, 71(11-12):1297–1313.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., et al. (2020). Array programming with numpy. *Nature*, 585(7825):357–362.

Helz, R. T. and Thornber, C. R. (1987). Geothermometry of kilauea iki lava lake, hawaii. *Bulletin of volcanology*, 49(5):651–668.

Herzberg, C. and O'hara, M. (2002). Plume-associated ultramafic magmas of phanerozoic age. *Journal of Petrology*, 43(10):1857–1883.

Hirschmann, M., Ghiorso, M., Davis, F., Gordon, S., Mukherjee, S., Grove, T., Krawczynski, M., Medard, E., and Till, C. (2008). Library of experimental phase relations (lepr): A database and web portal for experimental magmatic phase equilibria data. *Geochemistry, Geophysics, Geosystems*, 9(3).

Hollister, L. S., Grissom, G., Peters, E., Stowell, H., and Sisson, V. (1987). Confirmation of the empirical correlation of al in hornblende with pressure of solidification of calc-alkaline plutons. *American Mineralogist*, 72(3-4):231–239.

Iacovino, K., Matthews, S., Wieser, P. E., Moore, G., and Begue, F. (2021). Vesical: An open source thermodynamic model engine for mixed volatile solubility in silicate melts. *Available at EarthArxiv -https://doi.org/10.31223/X5D606*.

Johnson, M. (1988). Experimental calibration of an aluminum-in-hornblende geobarometer applicable to calc-alkaline rocks. *Eos*, 69:1511.

Lee, C.-T. A. and Anderson, D. L. (2015). Continental crust formation at arcs, the arclogite "delamination" cycle, and one origin for fertile melting anomalies in the mantle. *Science Bulletin*, 60(13):1141–1156.

Masotta, M., Mollo, S., Freda, C., Gaeta, M., and Moore, G. (2013). Clinopyroxene–liquid thermometers and barometers specific to alkaline differentiated magmas. *Contributions to Mineralogy and Petrology*, 166(6):1545–1561.

Matthews, S., Shorttle, O., and Maclennan, J. (2016). The temperature of the i celandic mantle from olivine-spinel aluminum exchange thermometry. *Geochemistry, Geophysics, Geosystems*, 17(11):4725–4752.

Matzen, A. K., Baker, M. B., Beckett, J. R., and Stolper, E. M. (2011). Fe–mg partitioning between olivine and high-magnesian melts and the nature of hawaiian parental liquids. *Journal of Petrology*, 52(7-8):1243–1263.

Molina, J., Moreno, J., Castro, A., Rodríguez, C., and Fershtater, G. (2015). Calcic amphibole thermobarometry in metamorphic and igneous rocks: New calibrations based on plagioclase/amphibole al-si partitioning and amphibole/liquid mg partitioning. *Lithos*, 232:286–305.

Mollo, S., Putirka, K., Misiti, V., Soligo, M., and Scarlato, P. (2013). A new test for equilibrium based on clinopyroxene–melt pairs: clues on the solidification temperatures of etnean alkaline melts at post-eruptive conditions. *Chemical Geology*, 352:92–100.

Montierth, C., Johnston, A. D., and Cashman, K. V. (1995). An empirical glass-composition-based geothermometer for mauna loa lavas. *Washington DC American Geophysical Union Geophysical Monograph Series*, 92:207–217.

Mutch, E., Blundy, J., Tattitch, B., Cooper, F., and Brooker, R. (2016). An experimental study of amphibole stability in low-pressure granitic magmas and a revised al-in-hornblende geobarometer. *Contributions to Mineralogy and Petrology*, 171(10):1–27.

Neave, D. A., Bali, E., Guðfinnsson, G. H., Halldórsson, S. A., Kahl, M., Schmidt, A.-S., and Holtz, F. (2019). Clinopyroxene–liquid equilibria and geothermobarometry in natural and experimental tholeiites: the 2014–2015 holuhraun eruption, iceland. *Journal of Petrology*, 60(8):1653–1680.

Neave, D. A. and Putirka, K. D. (2017). A new clinopyroxene-liquid barometer, and implications for magma storage pressures under icelandic rift zones. *American Mineralogist*, 102(4):777–794.

pandas development team, T. (2020). pandas-dev/pandas: Pandas.

Pritchard, M., Mather, T., McNutt, S. R., Delgado, F., and Reath, K. (2019). Thoughts on the criteria to determine the origin of volcanic unrest as magmatic or non-magmatic. *Philosophical Transactions of the Royal Society A*, 377(2139):20180008.

Pu, X., Lange, R. A., and Moore, G. (2017). A comparison of olivine-melt thermometers based on d mg and d ni: The effects of melt composition, temperature, and pressure with applications to morbs and hydrous arc basalts. *American Mineralogist*, 102(4):750–765.

Pu, X., Moore, G. M., Lange, R. A., Touran, J. P., and Gagnon, J. E. (2021). Experimental evaluation of a new h2o-independent thermometer based on olivine-melt ni partitioning at crustal pressure. *American Mineralogist: Journal of Earth and Planetary Materials*, 106(2):235–250.

Putirka, K. (1999). Clinopyroxene+ liquid equilibria to 100 kbar and 2450 k. *Contributions to Mineralogy and Petrology*, 135(2-3):151–163.

Putirka, K. (2016). Amphibole thermometers and barometers for igneous systems and some implications for eruption mechanisms of felsic magmas at arc volcanoes. *American Mineralogist*, 101(4):841–858.

Putirka, K., Johnson, M., Kinzler, R., Longhi, J., and Walker, D. (1996). Thermobarometry of mafic igneous rocks based on clinopyroxene-liquid equilibria, 0–30 kbar. *Contributions to Mineralogy and Petrology*, 123(1):92–108.

Putirka, K., Ryerson, F., and Mikaelian, H. (2003). New igneous thermobarometers for mafic and evolved lava compositions, based on clinopyroxene+ liquid equilibria. *American Mineralogist*, 88:1542–1554.

Putirka, K. D. (2005). Igneous thermometers and barometers based on plagioclase+ liquid equilibria: Tests of some existing models and new calibrations. *American Mineralogist*, 90(2-3):336–346.

Putirka, K. D. (2008). Thermometers and barometers for volcanic systems. *Reviews in mineralogy and geochemistry*, 69(1):61–120.

Rasmussen, D. J., Plank, T. A., Wallace, P. J., Newcombe, M. E., and Lowenstern, J. B. (2020). Vapor-bubble growth in olivine-hosted melt inclusions. *American Mineralogist: Journal of Earth and Planetary Materials*, 105(12):1898–1919.

Ridolfi, F. and Renzulli, A. (2012). Calcic amphiboles in calc-alkaline and alkaline magmas: thermobarometric and chemometric empirical equations valid up to 1,130° c and 2.2 gpa. *Contributions to Mineralogy and Petrology*, 163(5):877–895.

Ridolfi, F., Renzulli, A., and Puerini, M. (2010). Stability and chemical equilibrium of amphibole in calc-alkaline magmas: an overview, new thermobarometric formulations and application to subduction-related volcanoes. *Contributions to Mineralogy and Petrology*, 160(1):45–66.

Roeder, P. and Emslie, R. (1970). Olivine-liquid equilibrium. *Contributions to Mineralogy and Petrology*, 29(4):275–289.

Schmidt, M. W. (1992). Amphibole composition in tonalite as a function of pressure: an experimental calibration of the al-in-hornblende barometer. *Contributions to mineralogy and petrology*, 110(2-3):304–310.

Scruggs, M. A. and Putirka, K. D. (2018). Eruption triggering by partial crystallization of mafic enclaves at chaos crags, lassen volcanic center, california. *American Mineralogist: Journal of Earth and Planetary Materials*, 103(10):1575–1590.

Sisson, T. and Grove, T. (1993). Temperatures and h 2 o contents of low-mgo high-alumina basalts. *Contributions to Mineralogy and Petrology*, 113(2):167–184.

Sugawara, T. (2000). Empirical relationships between temperature, pressure, and mgo content in olivine and pyroxene saturated liquid. *Journal of Geophysical Research: Solid Earth*, 105(B4):8457–8472.

Toplis, M. (2005). The thermodynamics of iron and magnesium partitioning between olivine and liquid: criteria for assessing and predicting equilibrium in natural and experimental systems. *Contributions to Mineralogy and Petrology*, 149(1):22–39.

Walker, B. A., Klemetti, E. W., Grunder, A. L., Dilles, J. H., Tepley, F. J., and Giles, D. (2013). Crystal reaming during the assembly, maturation, and waning of an eleven-million-year crustal magma cycle: thermobarometry of the aucanquilcha volcanic cluster. *Contributions to Mineralogy and Petrology*, 165(4):663–682.

Wallace, P. J., Plank, T., Bodnar, R. J., Gaetani, G. A., and Shea, T. (2021). Olivine-hosted melt inclusions: A microscopic perspective on a complex magmatic world. *Annual Review of Earth and Planetary Sciences*, 49.

Wan, Z., Coogan, L. A., and Canil, D. (2008). Experimental calibration of aluminum partitioning between olivine and spinel as a geothermometer. *American Mineralogist*, 93(7):1142–1147.

Waters, L. E. and Lange, R. A. (2015). An updated calibration of the plagioclase-liquid hygrometer-thermometer applicable to basalts through rhyolites. *American Mineralogist*, 100(10):2172–2184.

Wells, P. R. (1977). Pyroxene thermometry in simple and complex systems. *Contributions to mineralogy and Petrology*, 62(2):129–139.

Wieser, P. E., Edmonds, M., Maclennan, J., Jenner, F. E., and Kunz, B. E. (2019a). Crystal scavenging from mush piles recorded by melt inclusions. *Nature communications*, 10(1):1–11.

Wieser, P. E., Vukmanovic, Z., Kilian, R., Ringe, E., Holness, M. B., Maclennan, J., and Edmonds, M. (2019b). To sink, swim, twin, or nucleate: A critical appraisal of crystal aggregation processes. *Geology*, 47(10):948–952.

Williams, M. J., Schoneveld, L., Mao, Y., Klump, J., Gosses, J., Dalton, H., Bath, A., and Barnes, S. (2020). pyrolite: Python for geochemistry. *Journal of Open Source Software*, 5(50):2314.

Winpenny, B. and Maclennan, J. (2011). A partial record of mixing of mantle melts preserved in icelandic phenocrysts. *Journal of Petrology*, 52(9):1791–1812.

Wood, B. J. and Banno, S. (1973). Garnet-orthopyroxene and orthopyroxene-clinopyroxene relationships in simple and complex systems. *Contributions to Mineralogy and Petrology*, 42(2):109–124.